

| L<br>Number | Hits | Search Text   | DB    | Time stamp          |
|-------------|------|---|-------|---------------------|
| -           | 1    | ("5974144").PN.   | USPAT | 2004/02/02<br>15:11 |
| -           | 111  | (713/166).CCLS.   | USPAT | 2004/01/30<br>16:22 |
| -           | 1812 | ((open unlock) with partition) and<br>((close lock) with partition)   | USPAT | 2004/02/02<br>15:12 |
| -           | 15   | ((open unlock) with partition) and<br>((close lock) with partition) and secure<br>and private                 | USPAT | 2004/02/02<br>15:27 |
| -           | 328  | ((open unlock) with partition) and<br>((close lock) with partition) and (secure<br>private) and @ad<=20000630 | USPAT | 2004/02/02<br>15:28 |



US005623637A

**United States Patent** [19]

Jones et al.

[11] Patent Number: **5,623,637**[45] Date of Patent: **Apr. 22, 1997**

[54] **ENCRYPTED DATA STORAGE CARD INCLUDING SMARTCARD INTEGRATED CIRCUIT FOR STORING AN ACCESS PASSWORD AND ENCRYPTION KEYS**

[75] Inventors: **Michael F. Jones**, Nashua, N.H.;  
**Arthur Zachai**, Swampscott, Mass.

[73] Assignee: **Telequip Corporation**, Hollis, N.H.

[21] Appl. No.: **651,205**

[22] Filed: **May 17, 1996**

**Related U.S. Application Data**

[63] Continuation of Ser. No. 161,854, Dec. 6, 1993, abandoned.

[51] Int. Cl.<sup>6</sup> ..... **G06F 12/14**

[52] U.S. Cl. .... **395/491; 395/430; 395/442; 395/833; 395/188.01; 380/23; 380/25**

[58] Field of Search ..... **380/23, 25, 4; 395/188.01, 430, 442, 490, 491**

**References Cited****U.S. PATENT DOCUMENTS**

5,068,894 11/1991 Hoppe ..... 380/23  
5,124,117 6/1992 Tatebayashi et al. .... 380/21

5,204,663 4/1993 Lee ..... 340/825.34  
5,293,424 3/1994 Holtey et al. .... 380/23  
5,307,411 4/1994 Anvret et al. .... 380/25  
5,341,428 8/1994 Schatz ..... 380/23  
5,347,580 9/1994 Moiva et al. .... 380/25  
5,379,344 1/1995 Larsson et al. .... 380/23  
5,428,685 6/1995 Kadooka et al. .... 380/25  
5,448,045 9/1995 Clark ..... 380/25

**OTHER PUBLICATIONS**

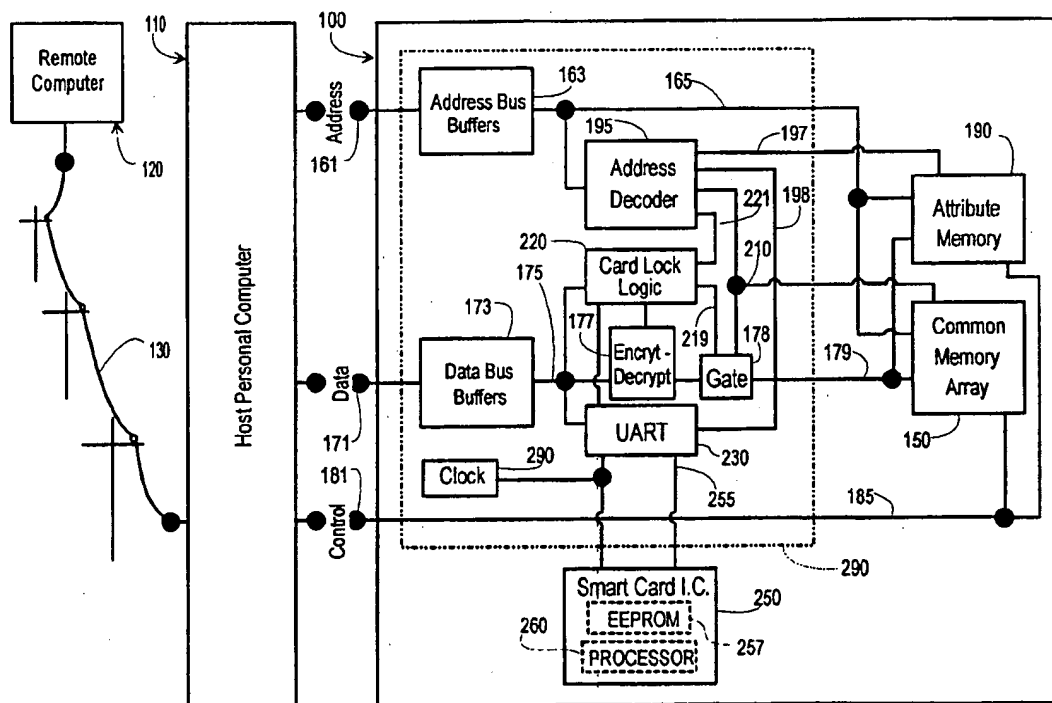
"Applied Cryptography" Bruce Schneier, John Wiley & Sons, Inc., 1994, pp. 219-243.

Primary Examiner—Reba I. Elmore

Attorney, Agent, or Firm—Banner & Witcoff, Ltd.

**[57]****ABSTRACT**

A detachable PCMCIA memory card incorporating a smartcard integrated circuit for storing a password value and logic circuitry for preventing access to information stored on the memory card unless the user of the host computer to which the memory card is connected can supply a password matching the stored password. The smartcard integrated circuit may also be used to store public and private key values used to encrypt and decrypt data stored on the card or elsewhere on the host computer or exchanged with a remote computer.

**3 Claims, 2 Drawing Sheets**

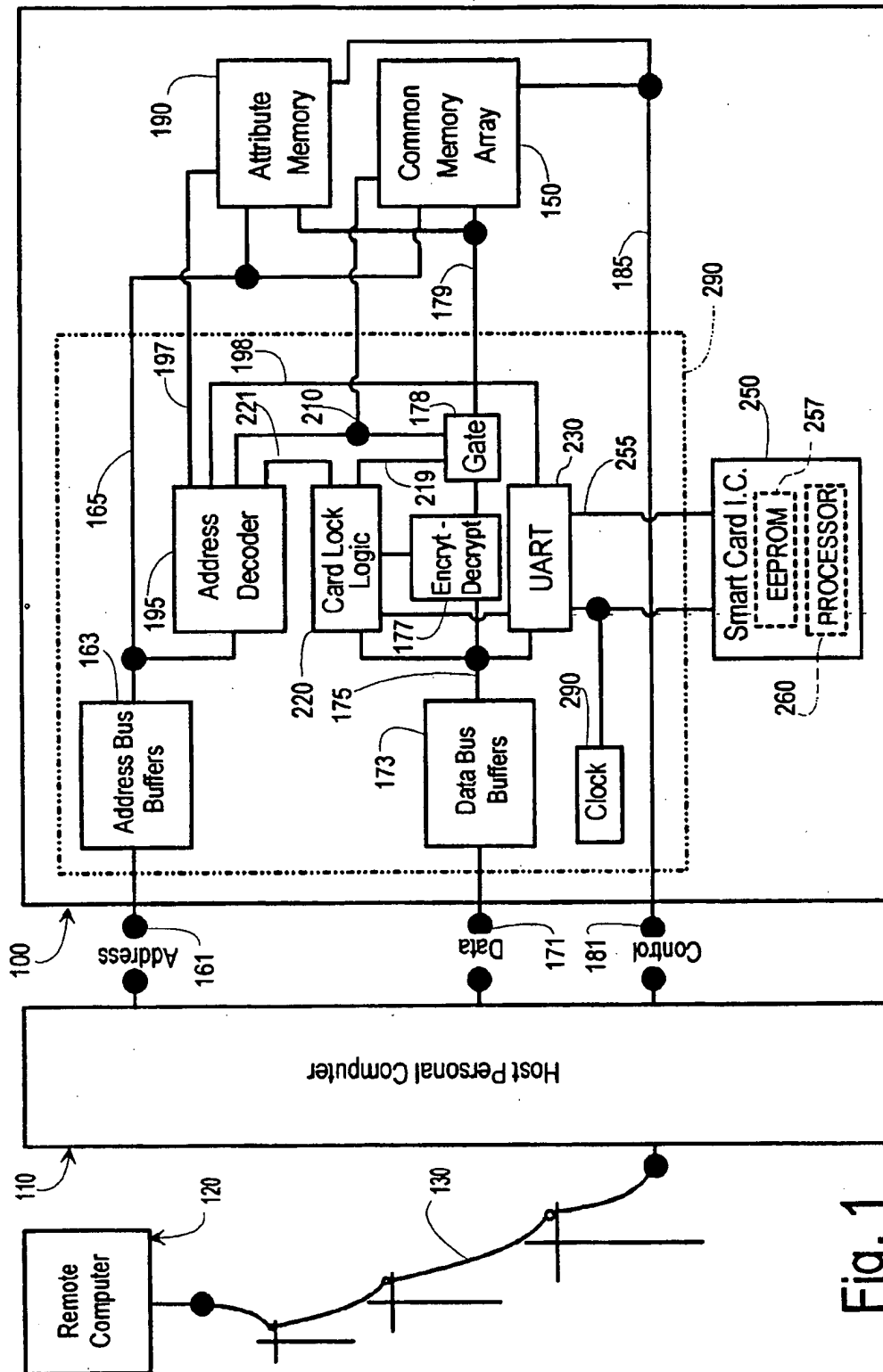
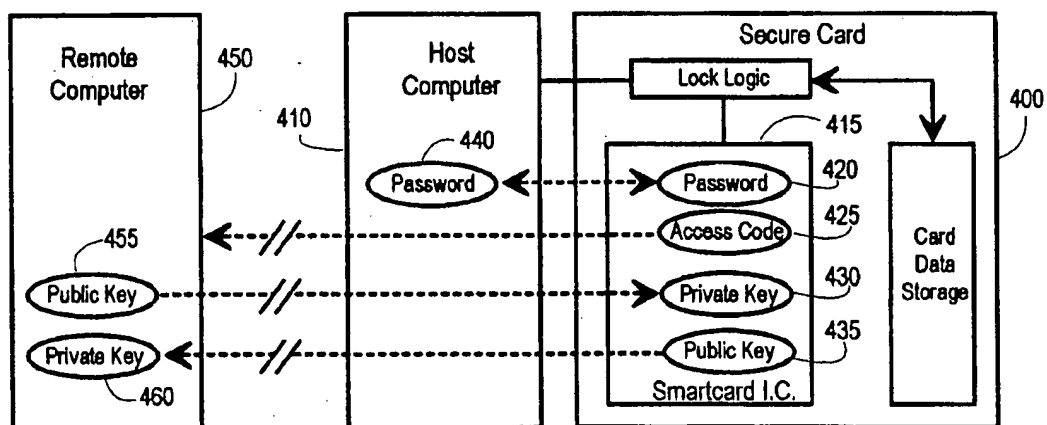
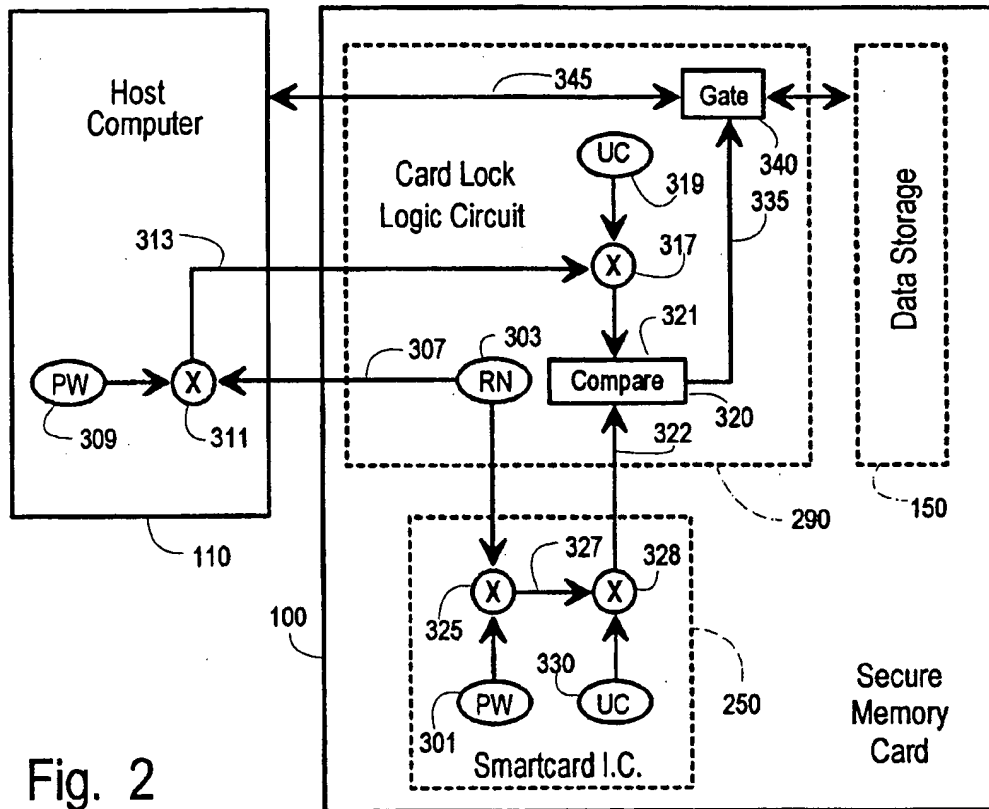


Fig. 1



1

# **ENCRYPTED DATA STORAGE CARD INCLUDING SMARTCARD INTEGRATED CIRCUIT FOR STORING AN ACCESS PASSWORD AND ENCRYPTION KEYS**

This application is a continuation of application Ser. No. 08/161,854 filed Dec. 6, 1993, abandoned.

## **FIELD OF THE INVENTION**

This invention relates generally to methods and apparatus for storing, processing and communicating private data.

## **BACKGROUND OF THE INVENTION**

Computers are widely used to store and process information which is considered private. For most businesses, the confidentiality of computer data is maintained using the practice followed for conventional business data: restricting access to office space where sensitive records are kept, whether those records take the form of documents kept in file cabinets or machine-readable data stored in an computer.

As the capabilities and usefulness of laptop and notebook computers have increased, functions formerly performed within the security of the office have moved to the field. Sales personnel and executives often travel with computers loaded with confidential data on pricing, customers, and strategic planning. Although available encryption and decryption programs can be used to protect such data when it is not in use, these programs are often inconvenient to use or provide poor security as a result of inadequate key management.

Encryption methods typically rely on "secret keys" known only to authorized users of the protected data. In the widely used Data Encryption Standard ("DES") developed and promulgated by the National Bureau of Standards, data is encyphered in 64-bit blocks using a single 56-bit key, as described in National Bureau of Standards' Federal Information Processing Standards Publication 46, "Data Encryption Standard," National Bureau of Standards (1977). Encryption techniques using two keys, one for encrypting the data and a different key for decryption, are called "public key" systems because the encryption key can be made public so that anyone can use the public key to encrypt sensitive data, but only a recipient with the secret key can decrypt it. One widely used and highly effective public key algorithm known as the "RSA" system, named after the inventors Rivest, Shamir and Adelman, is described in Rivest et al. U.S. Pat. No. 4,405,829.

The security of both single-key and public-key encryption systems depends on user's ability to keep the key or keys secret. Although both the DES and RSA encryption algorithms themselves can be depended upon to provide adequate security, neither system can safeguard data if the keys can be learned. The management of the keys themselves accordingly presents the most difficult component of good data security system.

## **SUMMARY OF THE INVENTION**

It is an object of the invention to securely store private information in a compact, easily transportable storage device which may be detached from the computer with which it is used.

It is still another object of the invention to protect such electronically stored data against unauthorized access when the detachable storage device is lost or stolen.

2

It is a further object of the present invention to provide a secure data storage device which may, at the option of the user, selectively limit access to all or part of the stored data using one or more passwords.

It is a related object of the invention to securely store access passwords, encryption or decryption keys, or digital signatures, in a tamper-proof substorage unit interconnected with a data access mechanism which are integral parts of a detachable computer memory card.

In a principle aspect, the present invention takes the form of a removable memory card, preferably implemented in conformity with the PCMCIA (Personal Memory Card Industry Association) interface standard, which provides the host computer to which it is connected with additional high-speed storage, the memory card consisting of a data storage unit, storage-access locking circuitry, and a tamper-proof key information substorage unit. In accordance with the invention, the locking circuitry is adapted to prevent access to the data stored on the memory card unless the would-be user first presents identifying information which is validated by the locking circuitry with reference to one or more key values stored in the key information substorage unit.

The removable memory card contemplated by the present invention allows data stored on the card to be made immediately available to the connected host computer upon proper presentation of a password known only to an authorized user. Once the password has been validated, the stored data may then be made available to the host processor in decrypted form.

In accordance with the invention, the key information substorage unit advantageously takes the form of a "smartcard" integrated circuit capable of storing secret key values which may be used to provide password-protected access to the data stored on the memory card, or optionally to provide secure storage for the encryption or decryption keys, or digital signatures, needed to allow the host computer to access and/or operate a secure information storage or telecommunications system. In accordance with the invention, access to data, passwords, digital signatures, or other key values stored on the memory card is limited to those who (1) have physical possession of the memory card and (2) knowledge of the memory card access password stored in the card's secure substorage unit.

The smartcard integrated circuit advantageously stores such passwords, public key and secret key values, and/or digital signatures in an Electrically Erasable Programmable Read Only Memory (EEPROM), and further includes its own microprocessor containing a stored program to allow reading and writing of the EEPROM through a serial I/O interface. The stored program within the smartcard IC allows an access password to be programmed into the EEPROM from an external source via the serial interface, but thereafter prevents that password value from being accessed. For enhanced security, the smartcard integrated circuit includes means for monitoring voltages and frequencies to detect abnormal conditions which may indicate an attempt to tamper with the key storage unit to gain unauthorized access to the stored secret key information.

These and other objects, features and advantages of the present invention will become more apparent by considering the following detailed description of a preferred embodiment of the invention, during which frequent reference will be made to the attached drawings.

## **BRIEF DESCRIPTION OF THE DRAWINGS**

FIG. 1 of the drawings is a block diagram of a secure memory card which embodies the principles of the inven-

tion, the memory card being shown interconnected with a host computer which is in turn connected to other computers by telecommunications links.

FIG. 2 is a data flow diagram depicting a preferred mechanism for providing password protection for information stored within a memory card of type shown in FIG. 1.

FIG. 3 is a data flow diagram illustrating the use of a secure data card as shown in FIG. 1 to protect the privacy of information being sent between a host computer and a remote computer.

#### DESCRIPTION OF THE PREFERRED EMBODIMENT

##### HARDWARE

As illustrated in FIG. 1 of the drawings, the preferred embodiment of the invention takes the form of a personal computer memory card indicated generally at 100. The memory card 100 is interconnected with a host computer 110 by means of a hardware and software interface which conforms to the Personal Computer Memory Card International Association (PCMCIA) standard which has been widely accepted for use in laptop and notebook computers. PCMCIA cards are commonly used to provide additional high-speed memory capacity to the connected host computer, or to implement fax and data modems, network access devices, and hard-disk mass storage devices. Type 1 PCMCIA cards have a form factor typically used to provide additional memory for data and application programs, while the thicker Type 2 cards are used to add telecommunications features and Type 3 cards are used for high-capacity hard disk drives that store up to 100 megabytes of data.

The removable character of PCMCIA storage devices can provide better data security than storage built into the computer itself, because the card may be detached from the computer and placed in a secure area when not in use. However, the cards themselves remain subject to possible theft or misuse. The embodiment of the invention shown in FIG. 1 provides significant additional security for data and programs stored in a detachable memory card by incorporating an access-locking mechanism for preventing access to the stored data by those who are unable to present an authorizing password.

The secure memory card 100 contemplated the invention is adapted to be connected via its PCMCIA interface to the host computer 110 which may in turn be connected to other computers by modem, or by a network, as illustrated by the connection of remote computer 120 via the telecommunications link 130 seen in FIG. 1.

The secure memory card 100 stores data in a common memory array 150, preferably implemented with non-volatile flash memory integrated circuits, enabling the common memory array to store 10 megabytes of data in an area small enough to be included on a credit-card sized Type I PCMCIA card. The data is stored in random access locations specified by address values supplied via the PCMCIA's standard 26-bit address bus terminals 161. The address terminals 161 provide address signals to an input address bus buffer circuit 163 which drives an internal address bus 165. Data transfers between the common memory array 150 and the host computer 110 are accomplished via the interface data terminals 171, a data bus buffer 173, an internal data bus 175, an internal encryption/decryption unit 177, a gate 178 and an internal data bus 179. Control signals are exchanged between the common memory array 150 and the host computer via the PCMCIA interface control terminals 181 and an internal control bus 185.

The address terminals 161, data terminals 171 and control terminals 181 seen in FIG. 1 are a simplified representation of the 68 pin PCMCIA standard interface which includes provision for 26 parallel address conductors (A0-A25), 16 parallel data conductors (D0-D15) and a remaining set of power and control conductors including power and ground connections and a collection of memory control signal connections (enable, select, wait, write, detect, etc.). The PCMCIA standard achieves interchangeability of cards of different functions by establishing standards for the physical card (dimensions and mechanical tolerances for the card and connectors), the card interface (pinout and signal definitions), and card software (which specifies the organization of data on the card and the record formats and protocols by which configuration information and data is exchanged with the host computer). Complete information which defines the PCMCIA standard is published by and available from the Personal Computer Memory Card International Association, 1030G East Duane Avenue, Sunnyvale, Calif. 94086. The present embodiment of invention conforms to the PC Card Standard Specification, Release 2.01, published in November, 1992.

To implement the PCMCIA interface standard, the secure memory card includes a non-volatile attribute memory 190 which stores information enabling the host computer to automatically identify the particular PCMCIA card as soon as the card and host are connected, and to automatically establish the appropriate hardware/software interface using suitable driver software which executes on the host computer 110.

The attribute memory 190 shares the internal address bus 165, data bus 175 and control bus 185 with the common memory array 150. An address decoder 195 monitors the address bus 165 and provides selection signals to the attribute memory 190 via an attribute memory enable line 197 when addresses within the address space of attribute memory 190 appear on address bus 165 concurrently with the activation of the Attribute Memory Select signal terminal -REG (not separately shown in FIG. 1) in the PCMCIA connector interface.

Similarly, the address decoder 195 selects the common memory array 150 whenever the address on address bus 165 is within the address space of array 150 by energizing a common memory enable line 210 which supplies an enable signal to the gate 178 in the data pathway to the common memory array 150.

Gate 178 prevents the common memory array 150 from exchanging data with the host 110 via data bus 179 unless an authorization signal is supplied to the gate 178 via a control line 219 from a card lock logic circuit 220. The card lock logic circuit 220 is connected to address decoder 195 via the lock enable line 221, permitting card logic 220 to identify addresses which designate memory locations in the common memory array 150 to which access may be denied under appropriate circumstances. The card lock logic circuit 220 is connected to the internal data bus 175 which provides a pathway for downloading memory access control commands from the host computer 110.

A smartcard input/output enable line 198 transmits an enable signal from the address decoder 195 to a Universal Asynchronous Receiver Transmitter (UART) 230 when information is to be transferred between the host computer 110 and a smartcard integrated circuit 250. The UART 230 is connected to the internal data bus 175 and operates to translate data received in bit-parallel form from data bus 175 into bit serial form for transfer to the smartcard integrated circuit "I.C." via its serial port 255.

The smartcard I.C. includes its own processor 260 and non-volatile EEPROM memory circuits 257 which operate as a secret key information substorage system. The processor 260 within the smartcard I.C. 250 is programmed to store secret key codes within the EEPROM 257, but to thereafter prohibit the stored secret keys from being accessed by any external interrogation. The smartcard I.C. may be implemented with a number of available devices, including the ST16F48 CMOU MCU-based Safeguarded Smartcard IC, with 8k EEPROM, available for SGS-Thomson Microelectronics, a member of the SGS-Thomson ST16XYZ family of devices, as specified in the SGS-Thomson Data Book (April, 1993). The ST16F48 includes an 8-bit processor, 288 bytes of RAM scratchpad storage, an 8k byte EEPROM data memory which forms the secure substorage unit, and a 16k byte program storage read-only memory for storing processing routines, including routines for processing and validating key values supplied to and read from the smartcard I.C. via the UART 230.

Data transfers and operations, both within the memory card 100 and between the card 100 and the host computer 110, are controlled by the card lock logic circuit 220. When the lock circuit enable line 221 is activated in response to the detection of an access control command address value by address decoder 195, the card lock logic circuit 220 responds to commands and data supplied to the internal data bus 175 from the host computer 110 via the data conductors 171 and the data bus buffers 173. The card lock logic circuit 220, the UART 230 and the smartcard I.C. 260 operate under the control of a common timing signal provided by an on-card clock generator circuit seen at 290 in FIG. 1.

The address space provided by the common memory array 150 is preferably partitioned into independently accessed regions. Each partition is specified in a Card Information Structure or "CIS" (to be described) which is stored in the attribute memory 190, and preferably corresponds to the memory space provided by one or more integrated circuits making up the array 150 such that a particular partition may be selected by the address decoder 195 which activates particular chip enable lines with the common memory enable output 210.

The access password itself is stored in the EEPROM 257 within the smartcard I.C. 250, the password storage operation being accomplished within the memory card 100 whenever a card lock logic activation address is supplied via address terminals 161 and the address buffer 163 to the address decoder 195 which in turn activates the card logic enable line 221. A password loading command applied via the data interface terminals 171 from the host computer is recognized by card lock logic 220 which channels the subsequent data sequence (the password itself) via the UART 230 and the serial port 255 of the smartcard I.C. 250 for storage at a predetermined location in the EEPROM 257.

Once a password has been stored for a particular partition, the card lock logic circuit 220 has exclusive control over access to that partition. Any attempt to access that partition (as detected by the address decoder 195) will be rejected, notifying the device driver software that a valid password must be provided. The driver software then prompts the user with a request for a valid password which, when entered, is sent via the data buffer 173 for validation. The card lock logic 220 routes the offered password to the smartcard I.C. with a request that it be compared with the password stored in the EEPROM 257. If the passwords match, the smartcard I.C. so notifies the card lock logic 220 which in turn notifies the device driver software executing in the host that the partition has been successfully unlocked. Thereafter, when

addresses within the unlocked partition are detected by the address decoder 195, the card lock logic will activate the gate 178 to permit data transfers between that partition and the data terminals 171.

To provide additional security, the data transferred over the 16-bit data bus between the data bus buffer 173 and the gate 178 is processed by the encryption-decryption unit 177 which preferably implements a symmetrical key algorithm, such as DES, based on a key value which stored in and fetched from the EEPROM 275 in the smartcard I.C. 250. The unit 250 encrypts data from the data bus buffer 173 prior to storing the data in the common memory array 150, and decrypts the data back into its original form when it is retrieved from the common memory array 150. This additional encryption mechanism protects data stored in the common memory array even if that data is successfully read from the flash memory chips making up the array 150. As discussed in more detail later, the secure key storage mechanism provided by the memory card may also be used to protect sensitive data being manipulated by mechanisms external to the memory card 100.

All of the operative circuitry making up the memory card 100, with the exception of the attribute memory 190, the common memory array 150, and the smartcard I.C. 250, is preferably implemented by means of a single, monolithic application specific integrated circuit (ASIC) as indicated within the dashed line rectangle 290 in FIG. 1. By integrating this circuitry in a monolithic integrated circuit, security against invasive attempts to ascertain built-in unlock codes (to be discussed) or to bypass or disable security functions, is substantially improved.

#### SOFTWARE

As previously noted, the attribute memory 190 stores information which specifies the nature of the memory card 100 and the format used for the information stored on the card. The attribute memory 190 holds a Card Information Structure ("CIS") which is organized in a "Metaformat" defined in Section 5 of the PCMCIA PC (Personal Computer) Card Standard, Release 2.01, for handling numerous different data recording formats. The CIS is organized as hierarchy of layers and takes the form of a chain (linked-list) of data blocks called "tuples" which begin at address 0 of the attribute memory 190.

The PCMCIA standard also establishes standards for the operation of host processor operating system software which can be used to simplify the design of specific device drivers which provide access to the memory card. The standard "Socket Services" and "Card Services" card interface software, when implemented on a given host computer, provides a Card Services interface with "Client Device Drivers," significantly simplifying the design of device drivers by providing much of the functionality required for communication with socketed PCMCIA cards. For host computers which are not provided with standard PCMCIA Card Services and Socket Services functions, the device driver directly interrogates the CIS structures in the attribute memory using standard link-list processing techniques, and provide direct software support for the bulk memory functions which would otherwise be supported by the PCMCIA Card Services interface.

Whether utilizing available Card Services routines or directly addressing and manipulating the memory card hardware interface, the device driver itself may be specified in the DOS CONFIG.SYS file and loaded when the host processor is initialized, or may take the form of an independently loadable TSR program. The discussion which follows describes the operation of a Client Device Driver adapted to

operate in conjunction with PCMCIA standard Card Service functions and notification mechanisms.

The programming interface to the PCMCIA Card Services software is defined in Section 3 of the PCMCIA Standard (Release 2.01) which specifies a variety of services which are available to Client Device Drivers, as well as callback mechanisms for notifying Client Device Drivers of status changes. In addition to conventional memory operations provided by Bulk Memory Service functions, the Card Services software also provides Client Utility functions which allow client device drivers to access and manipulate the CIS stored in the memory card's attribute memory 190. Card management routines, either forming a part of the Client Device Driver or part of a special purpose application program for configuring the memory card according to the users needs, are executed on the host computer. These card management routines in turn utilize the functions provided by the PCMCIA Card Services software to implement the following two special operations which not required for conventional PCMCIA memory cards:

#### PARTITION LOCK.

This operation accepts two parameters from the user: (1) a password value, typically taking the form of ASCIIZ (null-terminated string) of keyboarded characters entered by a user in response to a prompt, and (2) a partition identifier which specifies a portion of the address space provided by the common memory array 150. At the same time, the fact that a given partition has been locked, together with an identification of the EEPROM memory location of the password (but not its value) are recorded in the CIS entry for that partition.

The memory card 100 is initialized as a standard memory card before being first delivered to the end user, and provides one or more freely accessible storage partitions prior to receiving the first PARTITION LOCK command.

#### PARTITION UNLOCK.

The storage of a password associated with a particular password has the effect of locking that password against subsequent attempts to use the data or programs stored within that partition without first supplying a valid password.

Whenever a PCMCIA card is newly inserted into the socket of a running host computer, the Client Device Driver is notified by the Card Services software (via its CARD\_INSERTION callback function), so that it can process the card's CIS entries to identify each partition that may be password-protected. Similarly, when the host computer is first powered up and the Client Device Driver is initialized, the Client Device Driver calls Card Services functions to process the cards CIS entries to identify each partition that may be locked.

The device driver software then attempts to access each identified partition. If the partition is locked (as determined by the mechanism discussed above), the card lock logic 220 notifies the device driver of the locked condition, allowing the device driver to request a valid password from the user, either at the time the host computer is being initialized with an already socketed memory card, or at the time a memory card is first inserted into an already running host computer. Other Operations.

To support encryption and decryption systems, systems employing digital signatures, and secure telecommunications access protocols, examples of which will be discussed below, the card lock logic unit 220 and UART 230 also provide the capability for storing additional passwords, key values, access codes and the like in the secure substorage system provided by the smartcard I.C. 250, or alternatively

(but less securely) in the common memory array 150 or in the attribute memory 190.

#### PASSWORD AND KEY MANAGEMENT

A preferred mechanism for validating the user's password needed to unlock a particular memory partition is illustrated in FIG. 2 of the drawings. First, as previously described, the user who desires to protect information stored on the card supplies a secret password which is written into the smartcard I.C. memory as indicated at 301. When an attempt is made to access data protected by the secret password 301, the ASIC 290 implementing the card lock logic unit 220 generates a random number 303 which is supplied to the host computer 110 as indicated at 307. The host computer 110 then prompts the user to enter a password at 309. The offered password 309 is combined with the random number 303 at 311 and the result is returned at 313 to the ASIC 290. The returned value is then combined at 317 with a fixed unlock code 319 (built into the ASIC 290) to produce a final value which is applied to a first input 321 of a comparator 320.

At the same time, the random number 303 which was sent to the host is also sent to the smartcard I.C. 250 whose processor 260 is programmed to combine the random number 303 at 325 with the previously stored secret password 301 to form a result value at 327. The result value 327 is combined at 328 with a copy 330 of the unlock code 319, and the resulting final value is applied to the second input 322 of the comparator 320. If the final value at input 321 which is created by the password offered by the user matches the final value at input 322 created by the password stored within the smartcard I.C. 250, the partition associated with the stored password will be unlocked by sending an activation signal 335 to a data flow gate 340 connected in the path of a data bus 345 connecting the host computer 350 and the memory card's common memory array 360.

It is important to observe that the data stored in a protected partition within the memory card 100 is available only to those who possess both the card and the password. Neither possession of the card without knowledge of the password, nor knowledge of the password without physical possession of the card, will be sufficient to obtain access to the data.

The combined requirement that the bearer of the card also know the password can be used to provide security for data stored or transmitted outside the memory card, as well as for data stored within the card, as illustrated by the examples depicted in FIG. 3 of the drawings.

As depicted in FIG. 3, a secure memory card 400 is connected to a host computer 410 and includes a smartcard integrated circuit 415 which provides a secure substorage system which stores a password 420, an access code 425, a private key value 430, and a public key value 435.

The password 420 is used to verify the identity of the bearer of the memory card, who is required to enter of a valid password 440 when prompted by the host computer as previously discussed in conjunction with FIG. 2. Unless a valid password 440 is known to the bearer of the memory card 400, the additional codes and keys 425, 430 and 435 cannot be retrieved from the smartcard I.C. Requests for access to the a stored key is passed to the smartcard I.C. and processed by routines stored in the smartcard I.C. internal ROM program store, which denies any incoming request which is not accompanied by the submission of a valid password 440. However, if the bearer of the memory card 400 can provide a valid password 440 which matches the stored password 420, the following additional secure transactions are made possible between the host computer 410 and a remote computer 450.



First, the remote computer 450 may deny access to one or more of its capabilities unless a proper access code is transmitted from a would be remote station which establishes communications with the computer 450, typical via a dial-up telephone connection using modem transmission, or by a dedicated connection in a network or the like. By storing the needed access code in the a password-protected PCMCIA card, an improved mechanism is provided for establishing the authority of a remote caller to access designated functions in the remote computer. Preferably, the validity of the stored access code is established by a challenge-and-response exchange of the type illustrated in FIG. 2, in which the remote computer 450 transmits a challenge in the form of a random number which is combined with the stored access code 425 to form a response which is returned to the remote computer 450 for verification. In this way, the access code 425 is not transmitted and interception of either the challenge or response values by an intruder monitoring the exchange will not provide the intruder with the access code.

After the host computer 410 establishes its identity to the satisfaction of the remote computer 450 using the stored access code 425, the host computer then may request the transmission of encrypted data from the remote computer. To accomplish this, the two stations may use a two key encryption and decryption mechanism in which the remote computer 450 encrypts the data to be transmitted with a public key 455 and the host computer, using the private key value 430 stored in the smartcard I.C. 415, decrypts the received transmission to obtain the desired message text. Other computers in the network in possession of the public key 455 may also send secure transmissions to the host computer 410, but cannot decrypt such messages from others since only the possessor of the private key 430 is able to decrypt the transmissions.

Alternatively, the smartcard I.C. may store one or more public key values such as the public key 435 which enables the host processor to send secure transmissions to a selected remote receiving computer in possession of the corresponding private key 460. In the same way, single key encryption mechanisms such as DES, and systems requiring the identification of remote senders using digital signature techniques can be readily implemented using the password protected secure memory card as the mechanism for storing the needed key values.

The DES and RSA Encryption schemes, as well as numerous other systems for the secure information transmission and retention are described in detail in *Applied Cryptography: Protocols, Algorithms, and Source Code in C* by Bruce Schneier, John Wiley & Sons (1994), ISBN 0471597562. The use of the detachable, password protected secure memory card contemplated by the present invention provides additional security by providing an improved key management mechanisms which requires that a would-be user of such a system have both physical possession of the memory card (which holds the needed key values) and knowledge of the password which permits the key values to be accessed.

It is to be understood that the specific mechanisms and techniques which have been described are merely illustrative of an application of the principles of the invention. Numerous modifications may be made to the methods and apparatus described without departing from the true spirit and scope of the invention.

What is claimed is:

1. A removable memory card including external interface terminals for establishing data, address, control and power pathways between said card and corresponding socket terminals of a host computer, said memory card comprising:
  - a smartcard integrated circuit including a local processor and a substorage memory unit for storing a first password value and an encryption key value,
  - a non-volatile data storage memory,
  - gating means having a control input, said gating means being connected to establish a data pathway between said interface terminals and said data storage memory only when an authorization signal is applied to said control input, and
  - card lock logic means coupled to said smartcard integrated circuit for receiving a second password value from said host computer via said interface terminals, said card lock logic means including:
    - means for applying said authorization signal to said control input of said gating means only when said second password value bears a predetermined relation to said first password value stored in said substorage memory of said smartcard integrated circuit,
    - means for encrypting data transferred to said data storage memory via said gating means by combining data from said interface terminals with said encryption key value from said smartcard integrated circuit,
    - means for decrypting previously encrypted data transferred from said data storage means in response to said authorization signal by combining said encrypted data with said encryption key value stored in said substorage memory of said smartcard integrated circuit, and
    - means for preventing the transfer of said encryption key value from said substorage memory of said smartcard integrated circuit when said second password value does not bear said predetermined relation to said first password value.
2. A data processing system comprising, in combination,
  - a host computer including a interface connector and means for sending and receiving data via said interface connector, and
  - a removable memory card including:
    - interface terminals arranged to establish electrical connections to said interface connector of said host computer,
    - a data storage memory,
    - a smartcard integrated circuit including means for storing a first password value and a stored encryption key value,
    - an encryption and decryption unit connected between said interface terminals and said data storage memory for encrypting data being stored in said storage means by combining data received from said host computer with said encryption key value to produce stored encrypted data, and for thereafter decrypting said stored encrypted data by combining said encrypted data with said stored encryption key value,
    - gating means for controlling the transmission of data signals between said interface terminals and said data storage memory in response to an authorization signal, and
    - a card lock logic circuit comprising:
      - means for receiving a second password value from said host computer via said interface terminals,

## 11

detection means coupled to said smartcard integrated circuit and to said gating circuit for generating said authorization signal only if said second password value has a predetermined relationship with said first password value stored in said smartcard integrated circuit, and

means responsive to said authorization signal for transferring said stored encryption key value from said smartcard integrated circuit to said encryption and decryption unit.

3. A removable personal computer memory card for use with a connected personal computer, said card comprising, in combination:

a plurality of external data terminals for establishing a data pathway between said card and said personal computer,

plurality of external address terminals for establishing an address pathway between said card and said personal computer,

a non-volatile data memory having a data port and an address port,

circuit means connected to said address terminals for applying address signals received from said personal computer via said address pathway to enable the transfer of information to or from selected locations in said data memory via said data port,

encryption means connected between said external data terminals and said data port for converting data received from said personal computer via said data

## 12

pathway into encrypted data applied to said data port for storage in said selected locations in said data memory, said encryption means converting data in response to and in accordance with an encryption key value,

decryption means connected between said external data terminals and said data port for converting data transferred from said selected locations in said data memory into unencrypted data and for transferring said unencrypted data to said external data terminals, said decryption means converting data into said unencrypted data in response to and in accordance with said encryption key value,

a smartcard integrated circuit including a local processor and a secure substorage memory unit for storing a predetermined authorization value and said encryption key value,

first control means for accepting a password value from said personal computer and transferring said password value to said smartcard integrated circuit, and

means including a program executable by said local processor in said smartcard integrated circuit for comparing said password value with said authorization value and for transferring said encryption key value to said decryption means to enable said decryption means only when said password value and said authorization value have a predetermined relationship to one another.

\* \* \* \* \*



US005485439A

**United States Patent** [19][11] **Patent Number:** 5,485,439**Hamasaka et al.**[45] **Date of Patent:** Jan. 16, 1996

[54] **METHOD FOR  
RECORDING/REPRODUCING  
INFORMATION AND APPARATUS  
THEREFOR**

[75] **Inventors:** Hiroshi Hamasaka, Nishinomiya, Isao  
Sato, Neyagawa; Yoshihisa  
Fukushima, Osaka; Yuji Takagi,  
Hirakata; Yasushi Azumatani,  
Neyagawa, all of Japan

[73] **Assignee:** Matsushita Electric Industrial Co.,  
Ltd., Osaka, Japan

[21] **Appl. No.:** 785,296

[22] **Filed:** Oct. 30, 1991

[30] **Foreign Application Priority Data**

|               |      |       |          |
|---------------|------|-------|----------|
| Oct. 30, 1990 | [JP] | Japan | 2-293831 |
| Nov. 30, 1990 | [JP] | Japan | 2-337927 |
| Jul. 4, 1991  | [JP] | Japan | 3-164320 |

[51] **Int. Cl.<sup>6</sup>** G11B 7/00

[52] **U.S. Cl.** 369/47; 369/54; 369/58;  
369/50

[58] **Field of Search** 369/47, 50, 53,  
369/54, 58, 48; 395/425, 600; 360/27

[56] **References Cited**

**U.S. PATENT DOCUMENTS**

|           |        |                |          |
|-----------|--------|----------------|----------|
| 4,811,124 | 3/1989 | Dujari et al.  | 369/54 X |
| 5,136,569 | 8/1992 | Fennema et al. | 369/53 X |

**FOREIGN PATENT DOCUMENTS**

60-124727 7/1985 Japan

**OTHER PUBLICATIONS**

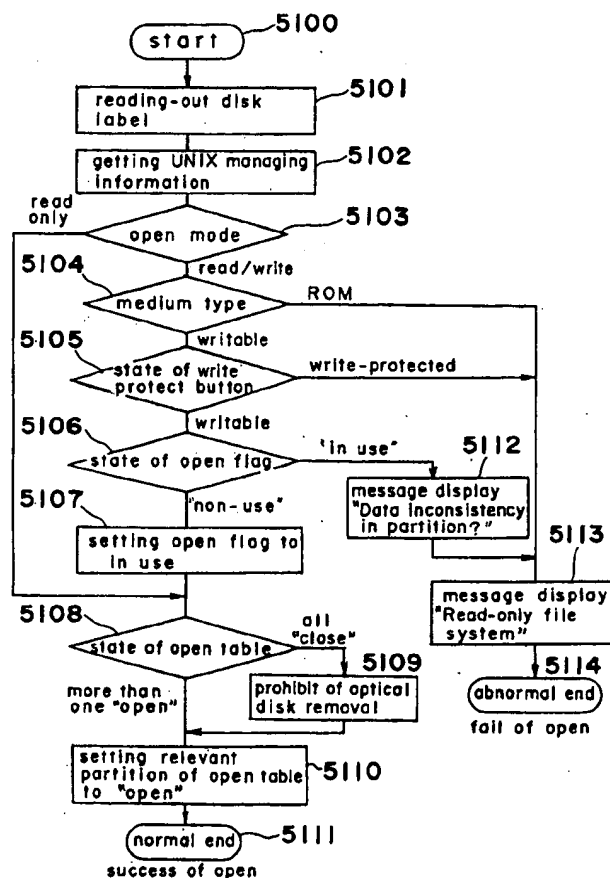
Apple Computer, Inc., Macintosh SE User's Manual, 1987  
(no month) pp. 126, 172, 180, 208, and 210.

*Primary Examiner*—W. R. Young  
*Attorney, Agent, or Firm*—Ratner & Prestia

[57] **ABSTRACT**

A method and apparatus for recording/reproducing information on an optical disk using the UNIX operating system. The optical disk has a plurality of partitions. An open table is provided to prevent the removal of the optical disk when the optical disk device is in use. The open table manages the use situation of the plurality of partitions. The optical disk is prohibited from being removed from the optical disk device when one partition is opened earlier than the other partitions. The removal prohibition of the optical disk is released when the final partition is closed.

3 Claims, 13 Drawing Sheets



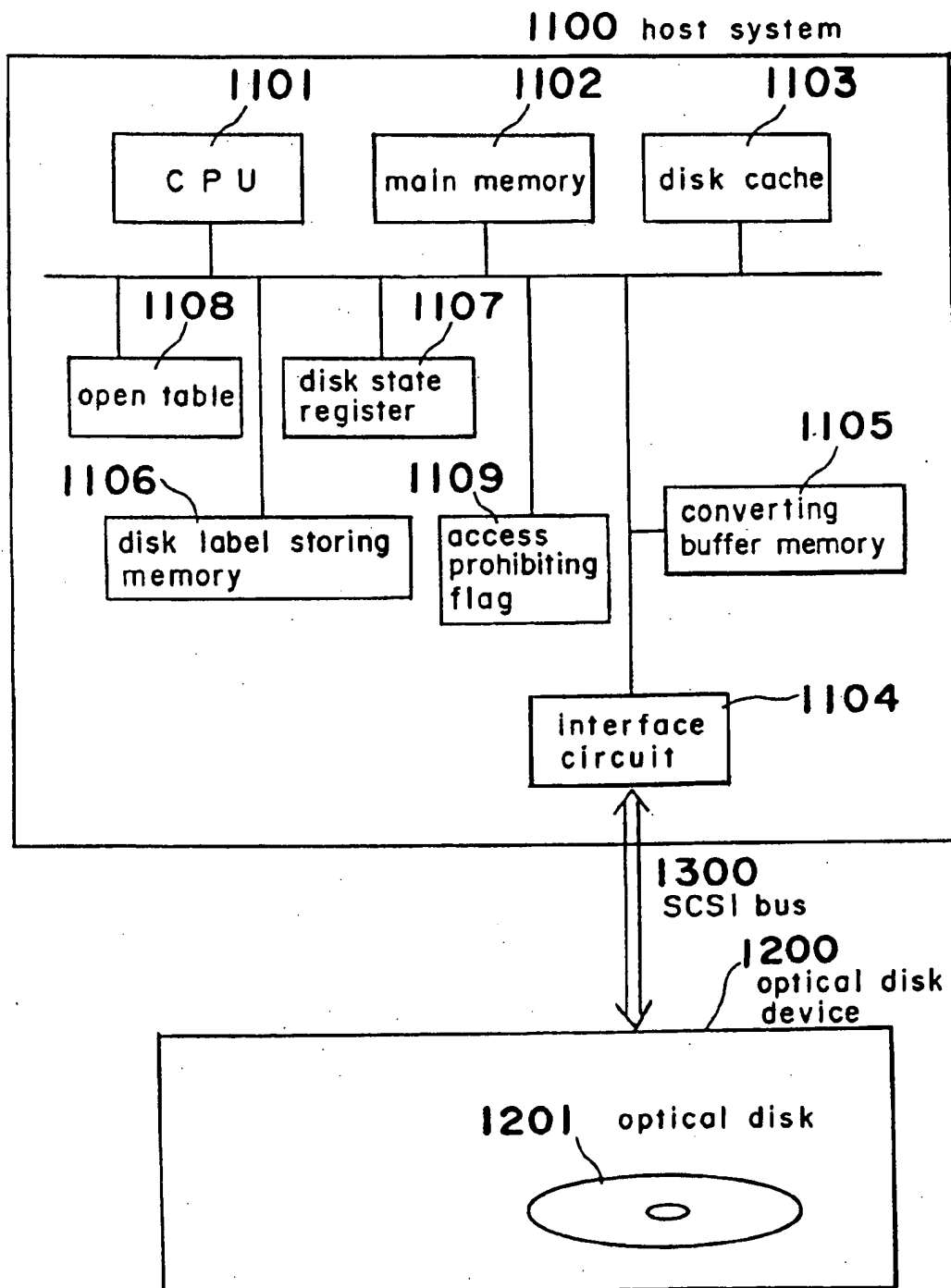
*Fig. 1*

Fig. 2(A)

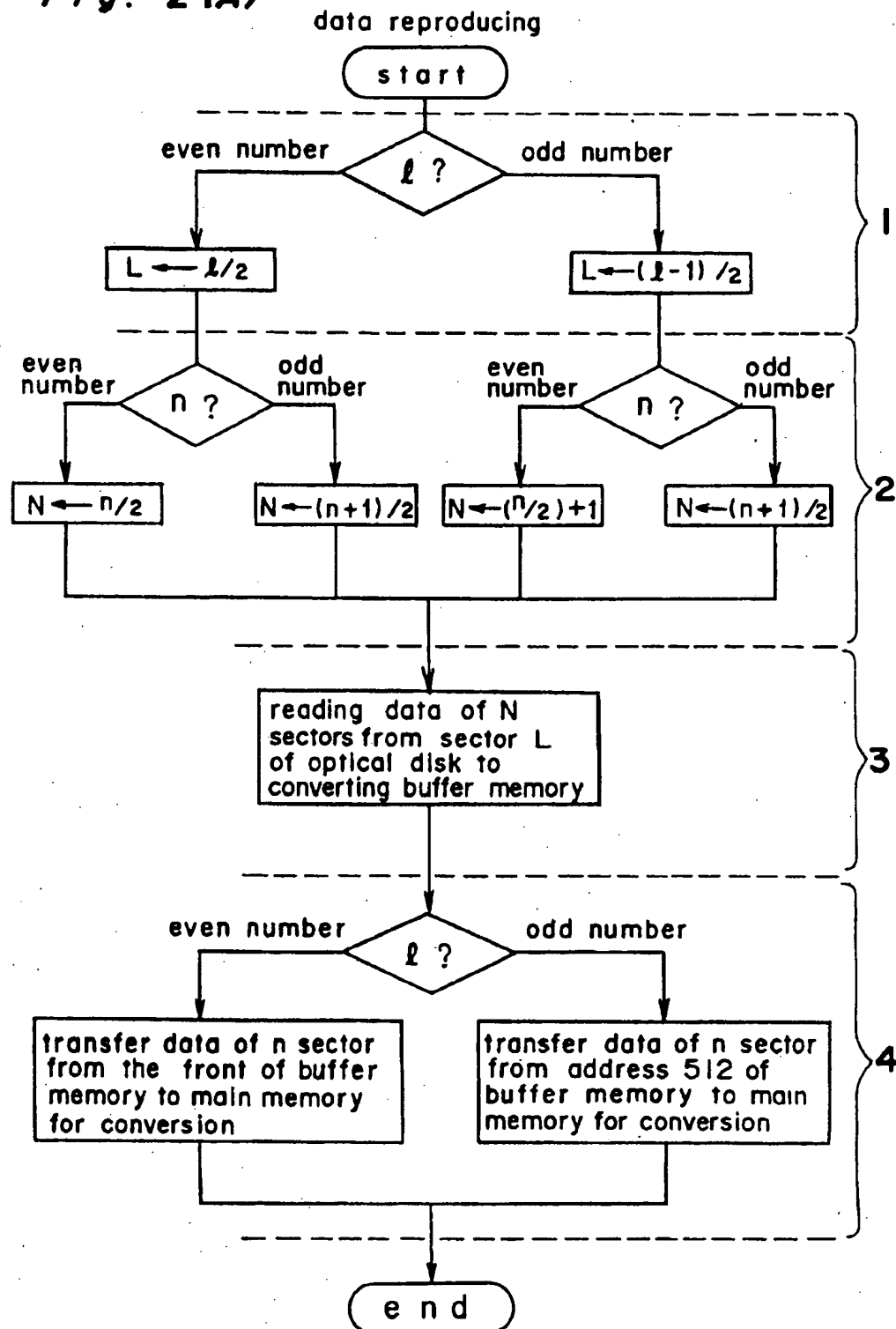
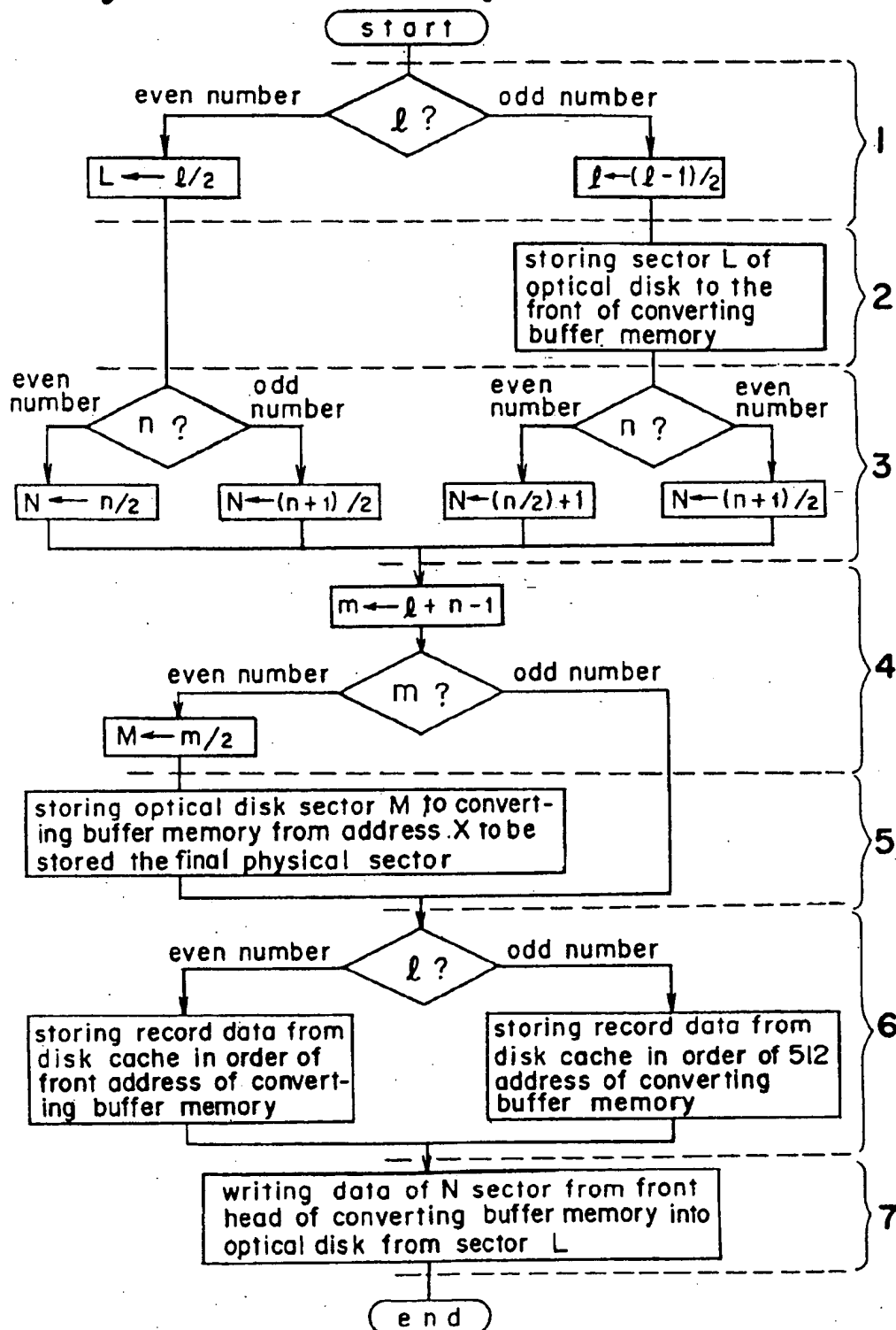
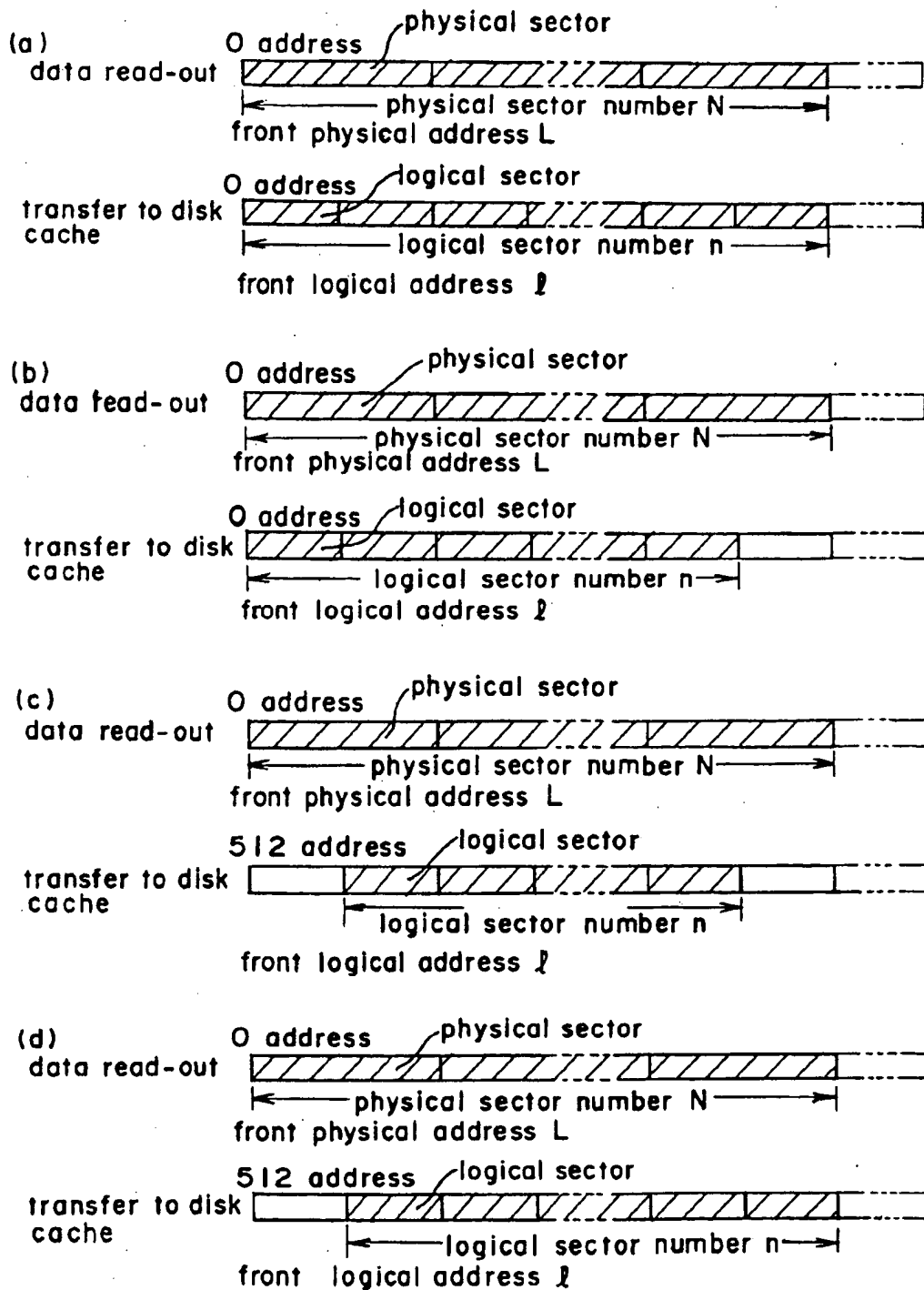
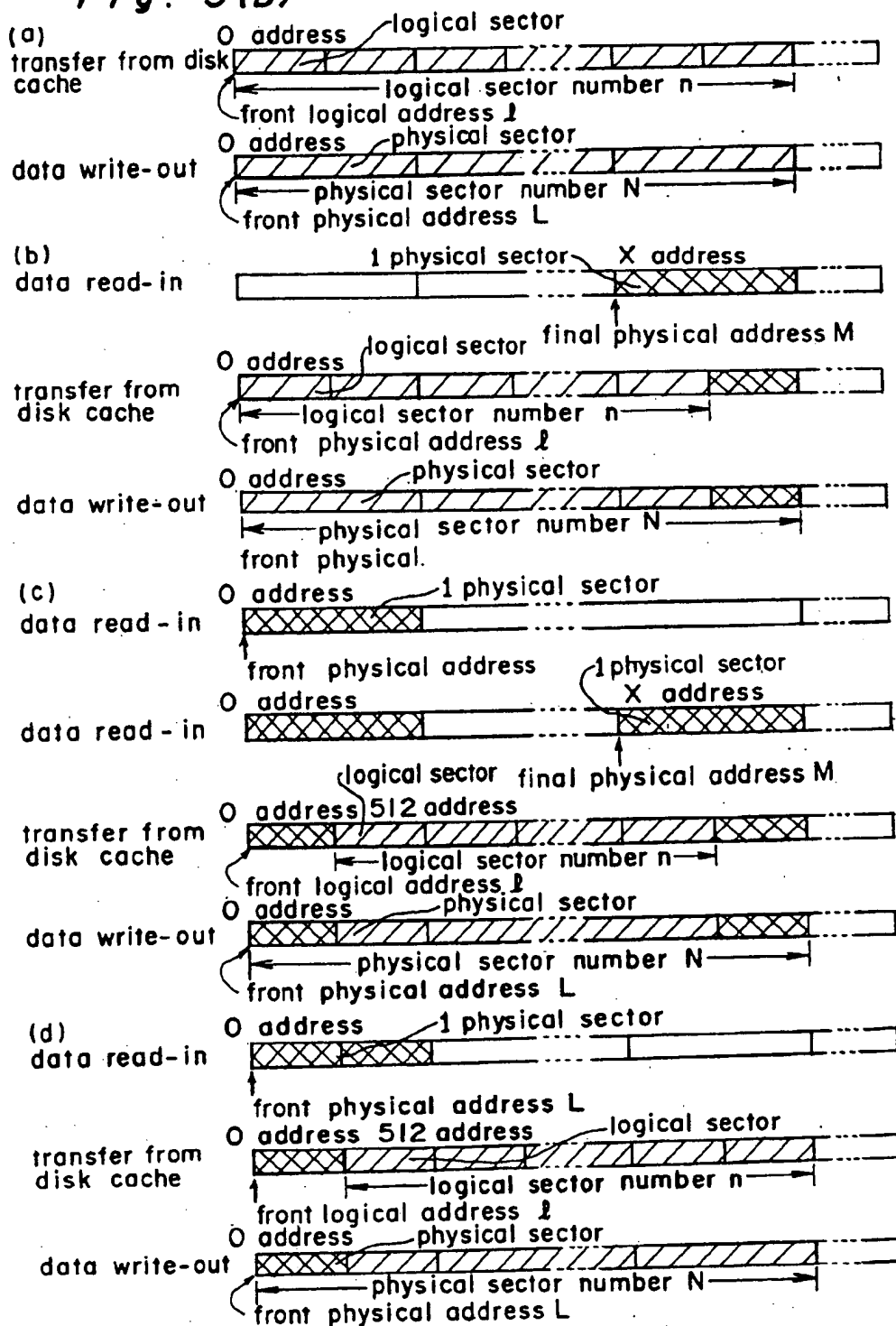


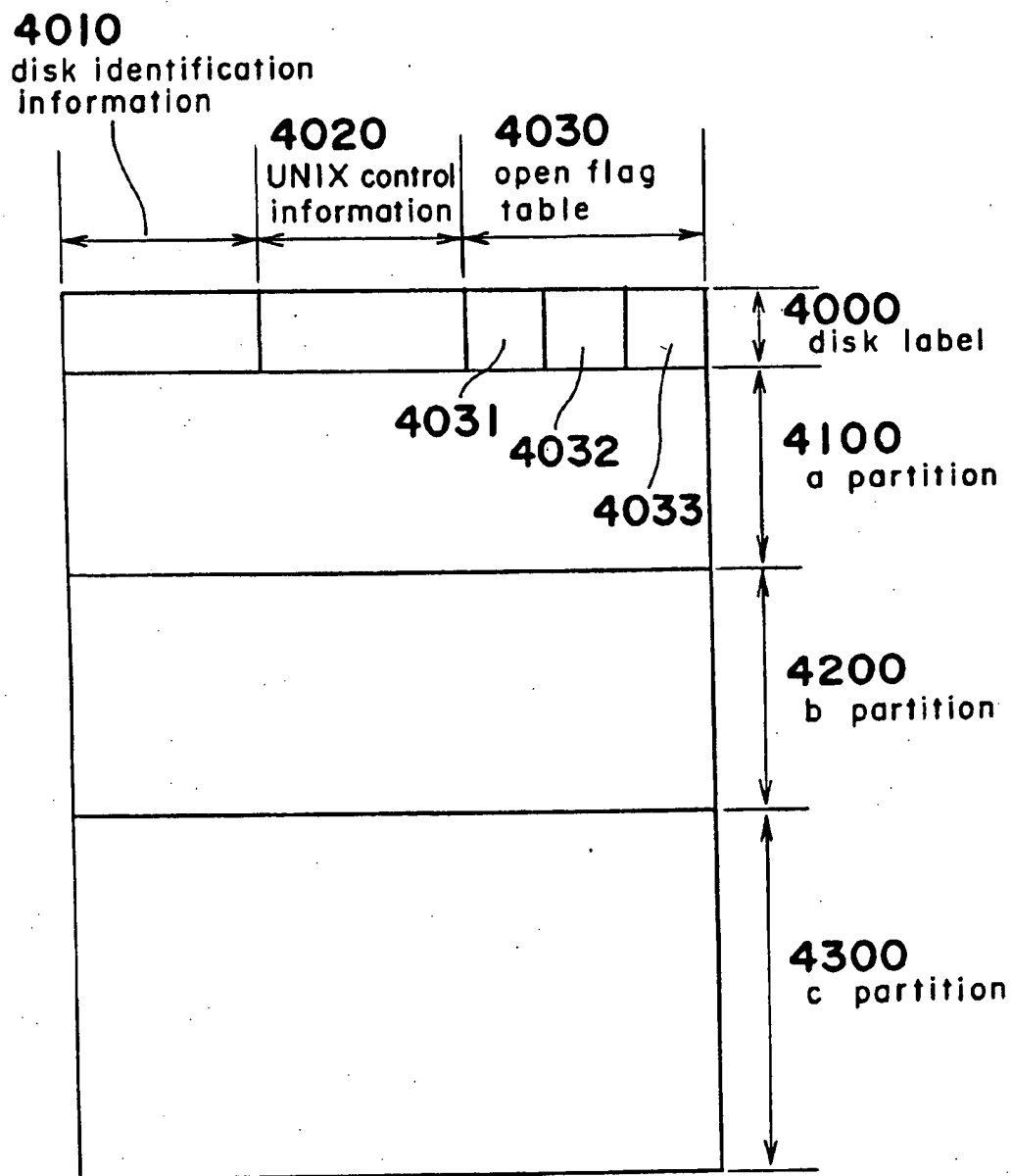
Fig. 2(B) data recording



*Fig. 3(A)*

**Fig. 3(B)**



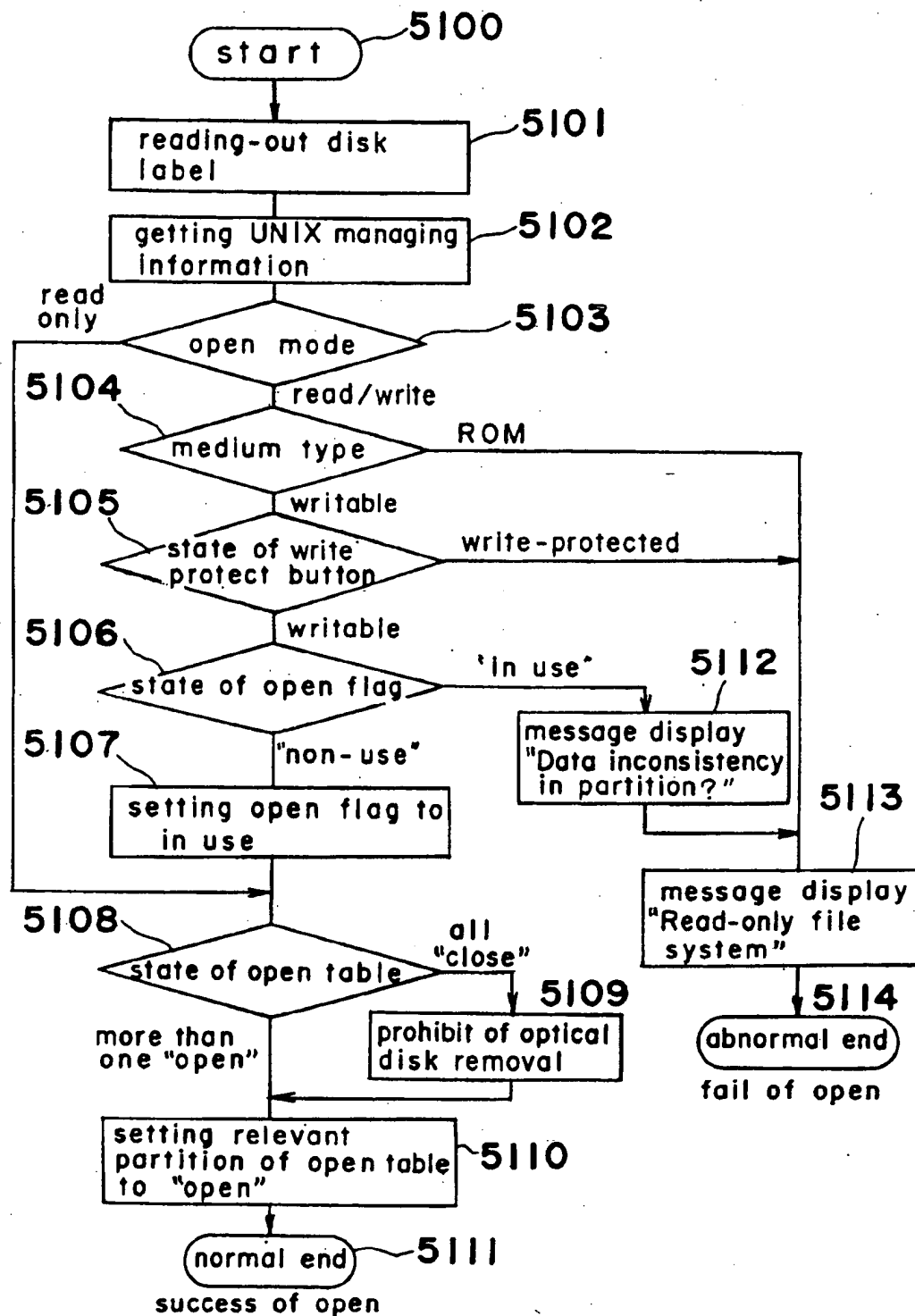
*Fig. 4*

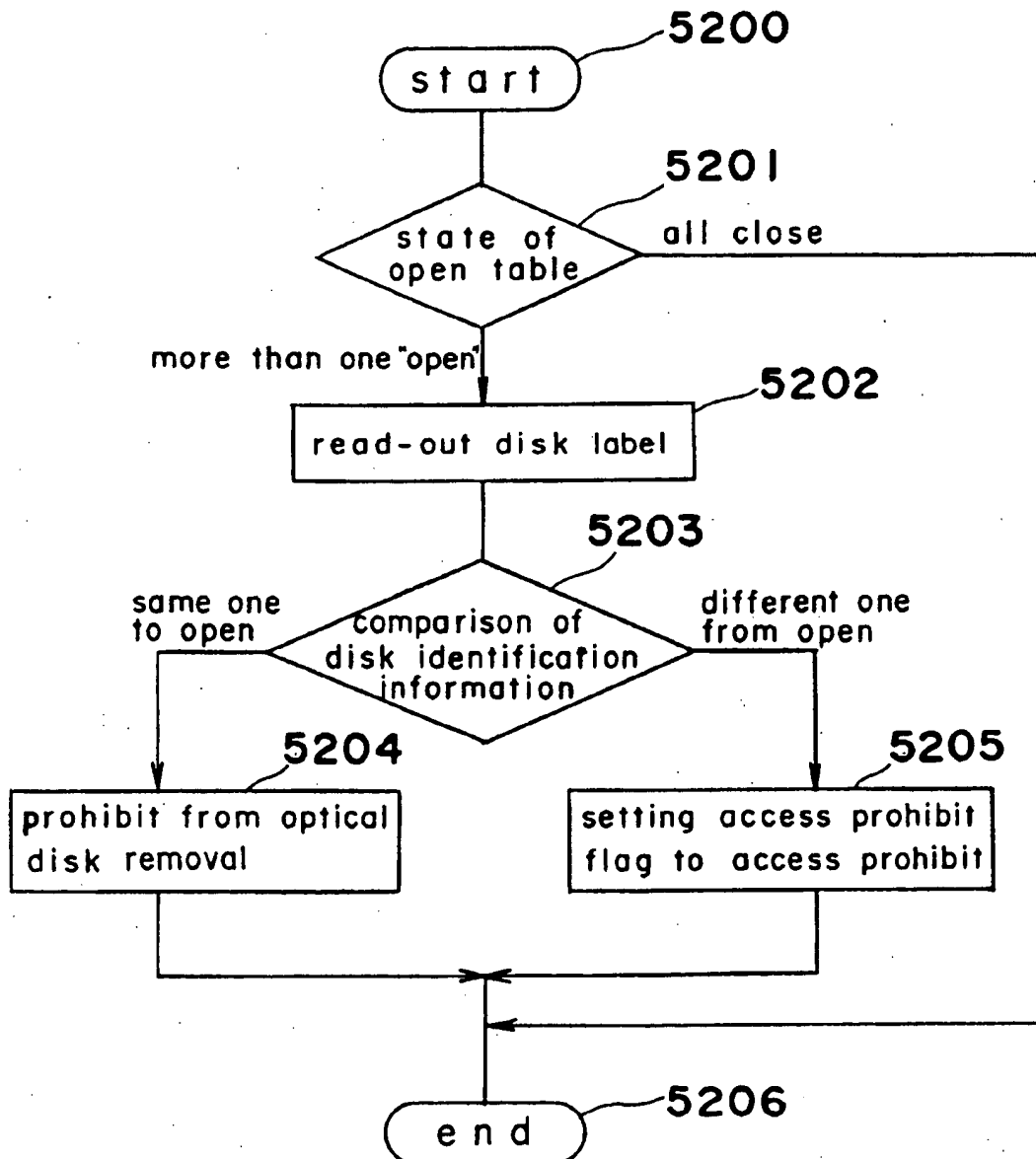
**4031** open flag of a partition

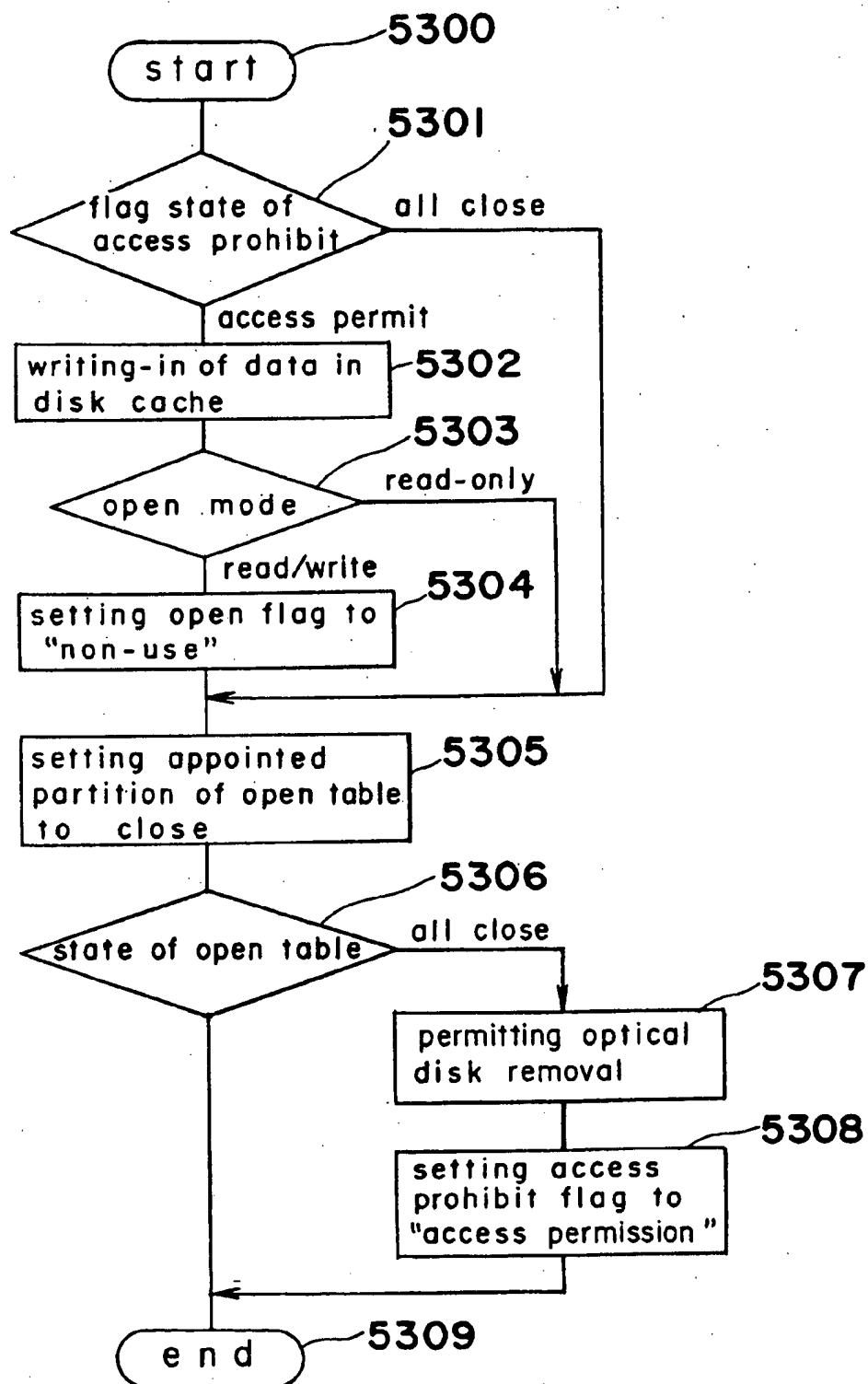
**4032** open flag of b partition

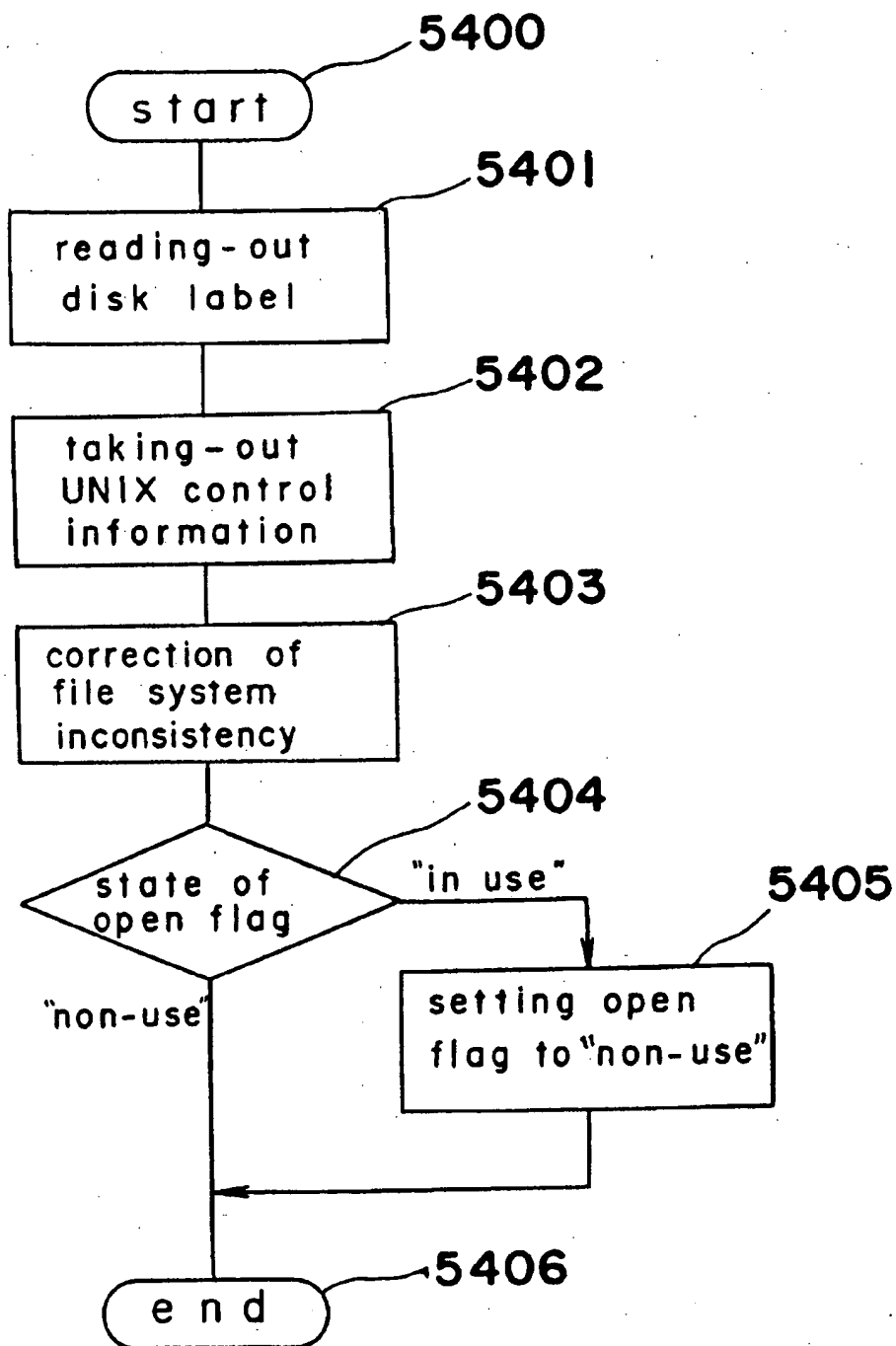
**4033** open flag of c partition

Fig. 5 (A)



*Fig. 5(B)*

*Fig. 5(C)*

*Fig. 5(D)*

**Fig. 6**

(a)

| a     | b     | c     |
|-------|-------|-------|
| close | close | close |

(b)

| a    | b     | c     |
|------|-------|-------|
| open | close | close |

(c)

| a    | b    | c     |
|------|------|-------|
| open | open | close |

(d)

| a     | b    | c     |
|-------|------|-------|
| close | open | close |

(e)

| a     | b     | c     |
|-------|-------|-------|
| close | close | close |

(f)

| a    | b     | c     |
|------|-------|-------|
| open | close | close |

(g)

| a     | b     | c     |
|-------|-------|-------|
| close | close | close |

**Fig. 7**

4031
4032
4033

(a)

|         |         |         |
|---------|---------|---------|
| a       | b       | c       |
| non-use | non-use | non-use |

(b)

|        |         |         |
|--------|---------|---------|
| a      | b       | c       |
| in use | non-use | non-use |

(c)

|        |        |         |
|--------|--------|---------|
| a      | b      | c       |
| in use | in use | non-use |

(d)

|         |        |         |
|---------|--------|---------|
| a       | b      | c       |
| non-use | in use | non-use |

(e)

|         |         |         |
|---------|---------|---------|
| a       | b       | c       |
| non-use | non-use | non-use |

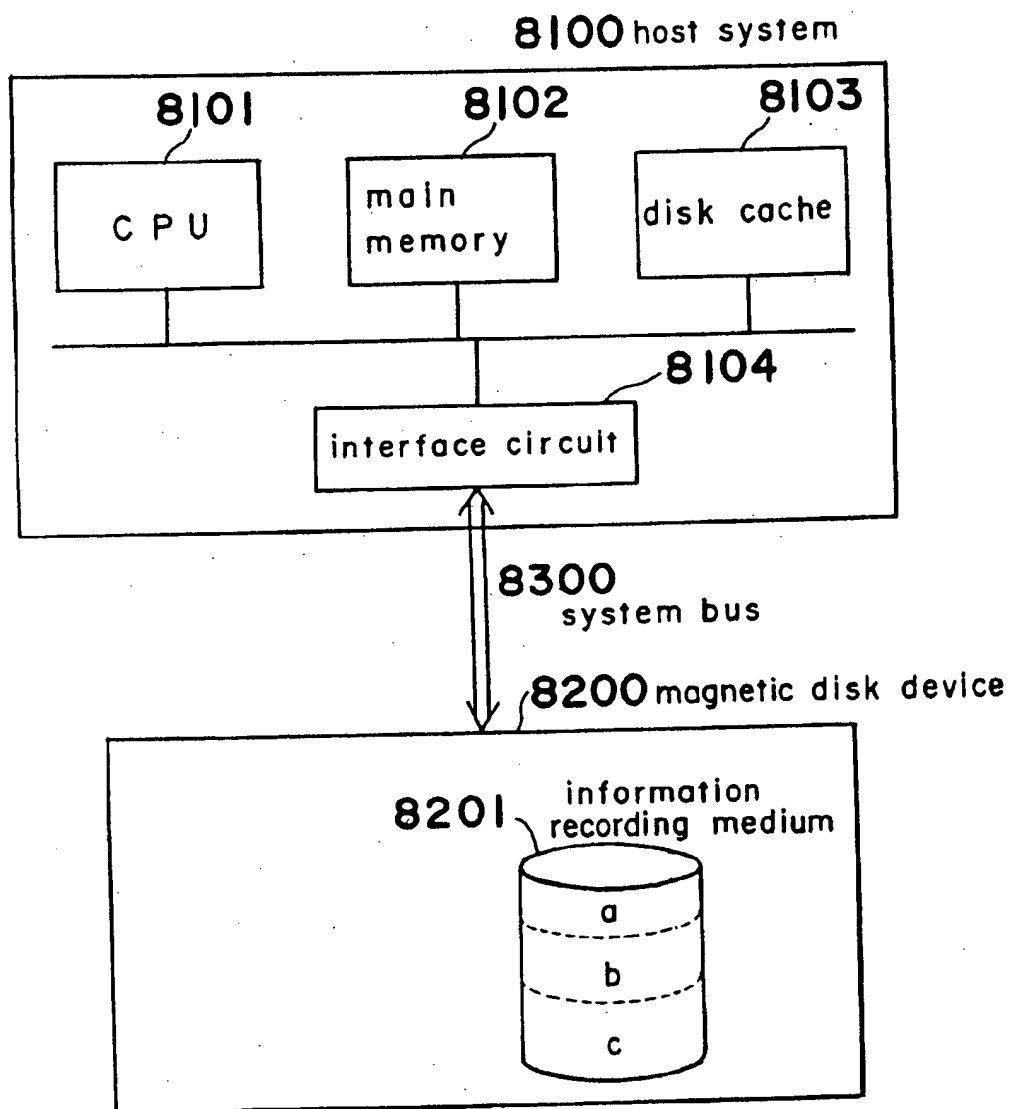
(f)

|        |         |         |
|--------|---------|---------|
| a      | b       | c       |
| in use | non-use | non-use |

(g)

|         |         |         |
|---------|---------|---------|
| a       | b       | c       |
| non-use | non-use | non-use |

**4031** open flag of a partition**4032** open flag of b partition**4033** open flag of c partition

*Fig. 8 PRIOR ART*



# 1

## METHOD FOR RECORDING/REPRODUCING INFORMATION AND APPARATUS THEREFOR

### BACKGROUND OF THE INVENTION

#### 1. Field of the Invention

The present invention relates to a method and an apparatus for recording/reproducing information in which an optical disk is used as an information recording medium.

#### 2. Description of the Prior Art

The operation of a conventional information recording reproducing apparatus is described below.

FIG. 8 is a construction view showing the conventional information recording reproducing apparatus, using UNIX as an operating system, in which a host system 8100 is connected with a magnetic disk device 8200 serving as an external memory via a system bus 8300.

The host system 8100 comprises a CPU 8101, a main memory 8102, a disk cache 8103, and an interface circuit 8104. The main memory 8102 stores a plurality of programs and data. The CPU 8101 reads out the data and the programs stored in the main memory 8102 into execution. The disk cache 8103 is a memory for making input/output processings at a high speed. The input and output of data between the host system 8100 and the external memory is carried out via the disk cache 8103. The interface circuit 8104 connects the host system 8100 to the system bus 8300.

The magnetic disk device 8200 accommodates an information recording medium 8201. The information recording medium 8201 is logically divided into three partitions a, b, and c and a file system is constructed in each partition.

The recording reproducing operation of the data in the conventional information recording reproducing of the above construction is described below.

The CPU 8101 reads out the program from the main memory 8102 into an execution. When the writing of data on the partition (a) of the information recording medium 8201 is necessary in this process, the CPU 8101 opens the partition (a) in read/write mode. In the open processing, the CPU 8101 reads out the management information required in controlling the file from the information recording medium 8201 via the interface circuit 8104.

Next, the CPU 8101 records data in the partition (a) of the information recording medium 8201. In the write processing of data, first, the CPU 8101 transfers data to be written to the disk cache 8103 from the main memory 8102.

When the CPU 8101 records the data from the main memory 8102 onto the information recording medium 8201, well-known write-back method is used. In this method, when data writing is required, the CPU 8101 stores data to be written into the information recording medium 8201 in the disk cache 8103. Then, the CPU 8101 writes the data stored in the disk cache 8103 into the information recording medium 8201 when a vacant region is necessary in the disk cache 8103, when the data is unnecessary in the host system 8100, or when a certain period of time has elapsed after the data is stored in the disk cache 8103. Meanwhile, when the read-out of the data is necessary, if data is on the cache the CPU 8101 reads out the data from the disk cache 8103, and data is not read out from the information recording medium 8201. If the data which is to be written on an address is on the disk cache, the CPU 8101 writes new data on the disk cache 8103. Therefore, the number of data writing on the

2

information recording medium 8201 is reduced. Thus, the number of inputs/outputs is reduced and the operation for recording/reproducing of data is performed at a high speed.

When the data writing is unnecessary, the CPU 8101 closes the partition (a). In the closing processing, the CPU 8101 issues an instruction of writing all data on the disk cache 8103 into the partition (a) of the information recording medium 8201 and at the same time, the partition (a) is logically cut off.

Thus, the data recording on the partition (a) of the information recording medium 8201 is carried out only from the opening processing of the partition (a) until the closing processing thereof.

Next, the data read-out of the host system 8100 is described below. When the data read-out from the (b) partition of the information recording medium 8201 is necessary, the CPU 8101 opens the (b) partition in read-only mode. The CPU 8101 reads out management information from the (b) partition of the information recording medium 8201 in the opening processing.

Next, the CPU 8101 reads out data from the (b) partition. In the read-out processing of data, the CPU 8101 reads out data from the (b) partition of the information recording medium 8201 in the magnetic disk device 8200 and issues an instruction of storing the data thus read out in the disk cache 8103 and at the same time, this data is transferred to the main memory 8102.

When the read-out of data is unnecessary, the CPU 8101 closes the (b) partition. In this closing processing, the CPU 8101 abandons the data on the disk cache 8103 and cuts off the (b) partition logically.

Thus, the reproduction of the data from the (b) partition is performed only from the opening processing of the (b) partition until the closing processing thereof. Any data is not written into the partition which has been opened in read-only mode.

The partition which has been opened is not opened again without being closed or the partition which has been closed is not closed again without being opened.

Let it be supposed that the (a) partition of the information recording medium 8201 is opened in read/write mode and data to be written into the (a) partition exists on the disk cache 8103. If the power source of the host system 8100 is cut off, the data on the disk cache 8103 is lost and the data to be written is not written into the (a) partition. Therefore, in the (a) partition, an inconsistency occurs in the file system. For example, the inconsistency occurs in the file control information or the difference occurs between the file control information and the actuality of the file.

If the CPU 8101 opens the (a) partition again in read/write mode and writes data therein with inconsistency occurring in the file system, a file which already exists may be destroyed. When the CPU 8101 detects the inconsistency of the file management information, the system goes down.

In order to avoid this, the CPU 8101 checks the consistency of the file system generated in each partition of the information recording medium 8201 when the power source is turned on, namely, in boot time. If an inconsistency is detected, the consistency of the file system is corrected. In the file system in which the inconsistency has been corrected thus, an existing file is not destroyed or the system does not go down due to the inconsistency in the file system.

When an optical disk which is a removable information recording medium is used in such a host system, the termination time of input/output in the program and the termina-

tion time of an operation for actual recording data into an information recording medium does not coincide with each other. Therefore, an optical disk may be taken out before the partition is closed, namely, before data on the disk cache is recorded. In particular, a plurality of programs simultaneously run in the UNIX. Therefore, there is a high possibility that the optical disk is taken out before all programs write data into the optical disk.

The inconsistency of the file system exists in the optical disk taken out before all data are written into the disk. Since the optical disk is exchangeable, it may be incorporated in the system or replaced after boot. Therefore, when the optical disk is used in the above system, the consistency is not corrected in boot time even when an inconsistency exists in the file system, and the existing file is broken or a system-down occurs due to the inconsistency of the file system.

When the optical disk is replaced before data on the disk cache is recorded, namely, when the optical disk is taken out and then another optical disk is inserted into, data which should have been already written on the optical disk is written into another optical disk replaced.

In the optical disk, since ROM (read-only memory) disk is used or in the write protect condition by the setting of a write protecting button, there is a case in which data cannot be written. When a user performs a data writing operation into the optical disk, the user does not write data on the optical disk. Therefore, inconsistency occurs between the data in the disk cache 8103 and the data on the optical disk, which may cause a system-down.

Since in a 4.3 BSD UNIX system in particular, a magnetic disk device having physical sector size as 512 B (bytes) is assumed to be an external memory, a data access is performed in units of 512 bytes. For example, in a manual of SunOs 4.0 of Sun Microsystem Corp. which is a representative manufacturer of a UNIX work stations, the sector size is 512 B in the item of System & Network Administration.

The optical disk is usually provided with a guide track which can be optically detected so that laser beams are irradiated on a recording layer of the guide track and a change which can be optically detected is made to record information. In a general optical disk to be used for processing information, a track formed on the optical disk is divided into a plurality of sectors of a fixed length so that information is recorded and reproduced on a sector by sector basis. In the process of manufacturing such an optical disk, simultaneously with the formation of a guide track, address information showing the position of a sector on the disk is formed for every constant interval. That is, in the process of manufacturing the optical disk, the size of the sector is physically determined. In ISO two kinds of formats of 512 B and 1024 B are determined as the format of the optical disk having a diameter of 130 mm. In the optical disk, in order to secure reliability, data is recorded with an error correcting code so as to correct an error and reproduce data. Since the error correcting code is formed with one sector being a unit, an error is corrected on a sector by sector basis. Therefore, in the case of an optical disk having a sector size of 1024 B, data cannot be recorded or reproduced with 512 B being a unit.

However, in an optical disk having a sector size as 1024 B, compared with an optical disk having a sector size as 512 B, smaller number of control information such as address suffices on a disk. Therefore, more user data can be recorded. As such, under a system in which UNIX is an OS, there is a demand of using an optical disk having a sector size of 1024 B.

But as described previously, an optical disk in which sector size is 1024 B cannot be used under UNIX which performs a data access on 512 B.

## SUMMARY OF THE INVENTION

Accordingly, an essential object of the present invention is to provide a method for recording/reproducing information by safely using an optical disk having a sector size of 1024 bytes and serving as a removable information recording medium under the use of UNIX operating system, and an apparatus for carrying out the method.

In a data reproducing operation, the method according to the present invention calculates the position of a sector corresponding to reproducing data existing on the information recording medium, reads all data of this sector into a converting buffer memory, calculates the position of data existing on the converting buffer memory and requested by a host system, and transfers data from the converting buffer memory to the host system with the position of the data thus calculated in response to the request made by the host system.

In a data recording operation, the method according to the present invention calculates the position of a sector of the information recording medium into which recording data is to be written. If the front of the recording data is not at the front of the sector, the sector is read out to be stored in the front of the converting buffer memory. If the end of the recording data is not at the final position of the sector, the sector is read out to be stored in the end of the converting buffer memory and the recording data is overwritten into the converting buffer memory from a position obtained by adding an offset corresponding to data existing in the range from the front of the sector to the front of the recording data to write the data on the converting buffer memory into the information recording medium.

In accordance with the above-described method, in the data reproducing operation, data read out from the information recording medium is stored in the converting buffer memory and a necessary portion of the data is transferred as reproducing data. In the data recording operation, data is read out from the information recording medium as necessary into the converting buffer memory and the data read out thus is written into the information recording medium with the recording data overwritten on the data read out. Thus, under a UNIX which executes a data access in units of 512 bytes, an optical disk having a sector size of 1024 bytes can be used by the use of the converting buffer memory which makes a conversion of the sector size.

According to the present invention, in order to safely use the optical disk serving as a removable medium, the following processing for controlling the medium is executed under UNIX which writes data according to write-back method.

The apparatus according to the present invention comprises an open table corresponding to the "open" state of each partition of the optical disk. The removal of the optical disk from the optical disk device is prohibited when any of partitions of the open table is in "open" state and permitted when all the partitions are in "closed" state. Thus, the optical disk can be taken out from the optical disk device only in the state where data has been correctly written.

According to the present invention, optical disks having different disk identification information are used to read out disk identification information from each optical disk for detecting a possibility that the optical disk has been

replaced. Then, it is decided whether or not the disk identification information thus read out is the same as information read when the partition has been opened. If both are the same, the removal of the optical disk from the optical disk device is prohibited. If both are different from each other, a data input to a replaced optical disk and a data output therefrom are prohibited. Thus, even though the optical disk device in use is reset and the optical disk becomes able to be taken out therefrom, processing continues by prohibiting the take-out of the optical disk again. In addition, even though the optical disk is forcibly replaced during its use, the data of the replaced optical disk is not destroyed.

Further, according to the present invention, the apparatus comprises an open flag, provided on the optical disk, corresponding to the use situation of each partition. The open flag is set to "in use" when the partition is opened in read/write mode and "non-use" when the partition is closed. If the open flag is in the state of "in use" in opening the partition again in read/write mode, the processing for opening the partition is decided as being unsuccessful. Since data is not written into a partition having an inconsistency in the file system due to the take-out of the optical disk during its use, an existing file is not destroyed or the system does not go down.

According to the method of the present invention, the processing for opening the partition is decided as being unsuccessful if it is detected that data writing into the optical disk is impossible. Since processing for writing data into an information recording medium is not erroneously executed, an inconsistency between data on the information recording medium and data on the disk cache does not occur. Thus, the system does not go down.

As described above, according to the present invention, the removable optical disk serving as the information recording medium can be safely used under the UNIX.

#### BRIEF DESCRIPTION OF THE DRAWINGS

These and other objects and features of the present invention will become clear from the following description taken in conjunction with the preferred embodiments thereof with reference to the accompanying drawings, in which:

FIG. 1 is a block diagram showing an apparatus for recording/reproducing information according to an embodiment of the present invention;

FIGS. 2A and 2B are flowcharts showing the flow of processing for converting a sector size in a method for recording/reproducing information according to an embodiment of the present invention;

FIGS. 3A(a-d) and 3B(a-d) are content views of a converting buffer memory according to an embodiment of the present invention;

FIG. 4 is a region view on an optical disk according to an embodiment of the present invention;

FIGS. 5A-5D are flowcharts showing the flow of processing for controlling a removable medium in the method for recording/reproducing information according to an embodiment of the present invention;

FIG. 6(a-g) is a content view showing an open table according to an embodiment of the present invention;

FIG. 7(a-g) is a content view showing an open flag table according to an embodiment of the present invention; and

FIG. 8 is a block diagram showing a conventional apparatus for recording/reproducing information.

#### DESCRIPTION OF THE PREFERRED EMBODIMENTS

Before the description of the present invention proceeds, it is to be noted that like parts are designated by like reference numerals throughout the accompanying drawings.

There is shown in FIG. 1 which is a block diagram of an information recording/reproducing apparatus using an optical disk according to an embodiment of the present invention. A host system 1100 and an optical disk device 1200 serving as an external memory are connected with each other via an SCSI bus (Small Computer System Interface) 1300.

The host system 1100 uses UNIX which performs a data access in units of 512 B (bytes) as an operating system. The host system 1100 comprises a CPU 1101, a main memory 1102, a disk cache 1103, an interface circuit 1104, a converting buffer memory 1105, a disk label storing memory 1106, a disk state register 1107, an open table 1108, and an access prohibiting flag 1109.

The main memory 1102 stores a plurality of programs and data. The CPU 1101 reads out data and programs stored in the main memory 1102 into execution. The UNIX has a data access to the main memory 1102 and the disk cache 1103. The disk cache 1103 is a memory for performing an input/output processing between the host system 1100 and an external memory at a high speed. The interface circuit 1104 is a circuit for connecting the host system 1100 to the SCSI bus 1300. The converting buffer memory 1105 is a work memory for converting a data access in units of 512 bytes demanded by the UNIX into units of sector size of 1024 bytes of an optical disk 1201. The disk label storing memory 1106 stores a disk label read out from the optical disk 1201. The disk label is described later. The disk state register 1107 is a register for storing the medium type of the optical disk 1201 and a write inhibition state of data set by means of a write protecting button. The open table 1108 holds the open state of each partition of the optical disk 1201. Each partition takes a value of "open" or "close". The access prohibiting flag 1109 keeping the prohibition state of a data input to the optical disk 1201 and a data output therefrom, thus taking a value of either "access prohibition" or "access permission".

The optical disk device 1200 accommodates the optical disk 1201 serving as an information recording medium. The optical disk device 1200 is provided with an SCSI serving as an interface, thus recording data into the optical disk 1201 and reproducing data therefrom, reporting the type of the optical disk 1201 and the protection state, prohibiting and permitting the removal of the optical disk 1201 from the optical disk device 1200 by means of an SCSI command. In addition, the optical disk device 1200 reports the Unit Attention state to the host system 1100 when the optical device is reset by the turn-on of the power source and the optical disk is inserted therein. The optical disk 1201 is accommodated in a cartridge (not shown) and a user can prohibit the writing of data to the optical disk by means of a write protection button (not shown) provided on the cartridge. The optical disk 1201 is logically divided into a plurality (three) of partitions, namely, an (a) partition, a (b) partition, and a (c) partition. A file system is constructed in each partition. The sector size of the optical disk 1201 is 1024 bytes.

The present invention using the optical disk having a sector of 1024 bytes under the UNIX in which a data access is executed in units of 512 bytes is described below with reference to FIGS. 2 and 3.

FIGS. 2A and 2B are flowcharts showing the flow of processing for converting a sector size in the method for

recording/reproducing information according to this embodiment.

FIGS. 3A and 3B are content views of the converting buffer memory 1105 for describing the embodiment. FIG. 3A shows the content of the converting buffer memory 1105 in a data read-out. FIG. 3B shows the content of the converting buffer memory 1105 in a data writing.

A sector in the information recording medium, the sector size of which is 512 bytes is called a logical sector; the address of the sector is called a logical address; the address of the front sector of data is called a front logical address; the address of the last sector of the data is called a final logical address; and the number of sectors in which data exist is called the number of logical sectors. The sector in information recording medium, the sector size of which is 1024 bytes is called a physical sector; the address of the sector is called a physical address; the address of the front sector of data is called a front physical address; the address of the last sector of the data is called a final physical address; and the number of sectors in which data exist is called the number of physical sectors.

In this embodiment, the addresses of all sectors and the address of the buffer memory are integers starting from "0", and the number of logical sectors and the number of physical sectors are integers of "1" or more. The converting buffer memory 1105 has a sufficient memory capacity for storing data to be recorded and reproduced.

The processing to be executed by the CPU 1101 in the case where the host system 1100 reads data from the optical disk 1201 accommodated in the optical disk device 1200 to the disk cache 1103 is described below with reference to FIGS. 1 and 2A.

Let it be supposed that the read-out of the data of the front logical address (l) and the logical sector number (n) becomes necessary when the host system 1100 is reading out the program from the main memory 1102 into execution. In the UNIX which performs a data access in units of 512 bytes, the data address in the information recording medium is designated by front logical address (l), and the number of blocks in the information recording medium is designated by the logical sector number (n).

The CPU 1101 executes the following processing as shown in FIG. 2A below in reading out the data of the logical sector number (n) from the front address (l).

(1) Calculation of front physical address (L) from front logical address (l):

If the front logical address (l) is an even number,

$$L=l/2$$

If the front logical address (l) is an odd number,

$$L=(l-1)/2$$

(2) Calculation of physical sector number (N) from logical sector number (n):

If the front logical address (l) is an even number and the logical sector number (n) is an even number,

$$N=n/2$$

If the front logical address (l) is an even number and the logical sector number (n) is an odd number,

$$N=(n+1)/2$$

If the front logical address (l) is an odd number and the logical sector number (n) is an even number,

$$N=(n/2)+1$$

If the front logical address (l) is an odd number and the logical sector number (n) is an odd number,

$$N=(n+1)/2$$

(3) Read-out of data from optical disk 1201 to converting buffer memory 1105:

The CPU 1101 drives the interface circuit 1104 to output an instruction of reading out the data of the physical sector number (N) from the front physical address (L) to the optical disk device 1200, reads the data having a capacity corresponding to the physical sector of a physical sector number (N) from the front address (L) of the optical disk 1201, and stores this data in the converting buffer memory 1105 from "0" address which is the front address thereof.

(4) Transfer of data of converting buffer memory 1105 to disk cache 1103:

If the front logical address (l) is an even number, the CPU 1101 transfers data having a capacity corresponding to the logical sectors of the logical sector number (n) from the front address of the converting buffer memory 1105 to the disk cache 1103.

If the front logical address is an odd number, the CPU 1101 transfers data having a capacity corresponding to the logical sectors of the logical sector number (n) from "512" address of the converting buffer memory 1105 to the disk cache 1103. "512" address is an address obtained by adding an offset corresponding to the capacity of one logical sector to the front address.

Then, the CPU 1101 transfers the data of the disk cache 1103 to the main memory 1102.

The content of data to be stored in the converting buffer memory 1105 in the data read-out from the optical disk 1201 is described below.

If the front logical address (l) and the logical sector number (n) are both even numbers, data is stored in the converting buffer memory 1105 as shown in FIG. 3A (a).

Data having a capacity corresponding to the physical sectors of the physical sector number (N) read from the front physical address (L) of the optical disk 1201 is stored in the converting buffer memory 1105 from "0" address thereof. All data stored thus are transferred from "0" address of the converting buffer memory 1105 to the disk cache 1103 as the data of the front logical address (l) and the logical sector number (n).

If the front logical address (l) is an even number and the logical sector number (n) are an odd number, data is stored in the converting buffer memory 1105 as shown in FIG. 3A (b).

Data having a capacity corresponding to the physical sector of the physical sector number (N) read from the front physical address (L) of the optical disk 1201 is stored in the converting buffer memory 1105 from the front "0" address thereof. Of the data stored thus, only data having a capacity corresponding to the logical sectors of the logical sector number (n) is transferred from "0" address of the converting buffer memory 1105 to the disk cache 1103 as the data of the front logical address (l) and the logical sector number (n).

If the front logical address (l) is an odd number and the logical sector number (n) is an even number, data is stored in the converting buffer memory 1105 as shown in FIG. 3A (c).

Data having a capacity corresponding to the physical sectors of the physical sector number (N) read from the front physical address (L) of the optical disk 1201 is stored in the converting buffer memory 1105 from the front "0" address thereof. Of data stored thus, data having a capacity corresponding to the logical sectors of the logical sector number

(n) is transferred from "512" address of the converting buffer memory 1105 to the disk cache 1103 as the data of the front logical address (l) and the logical sector number (n). "512" address is obtained by adding an offset corresponding to a capacity of one logical sector to "0" address.

If the front logical address (l) and the logical sector number (n) are both odd numbers, data is stored in the converting buffer memory 1105 as shown in FIG. 3A (d).

Data having a capacity corresponding to the physical sectors of the physical sector number (N) read from the front physical address (L) of the optical disk 1201 is stored in the converting buffer memory 1105 from "0" address thereof. Of stored data, data having a capacity corresponding to the logical sectors of the logical sector number (n) are transferred from "512" address to the disk cache 1103 as the data of the front logical address (l) and the logical sector number (n). "512" address is obtained by adding an offset corresponding to a capacity of one logical sector to "0" address.

As described above, according to the present invention, a request for reading out data is converted, the data is read out from the optical disk 1201 to the converting buffer memory, and only a necessary portion of read-out data is transferred to the disk cache 1103. Thus, the optical disk 1201, the sector size of which is 1024 bytes can be used in the operating system in which data is read out in units of 512 bytes.

Next, the processing to be executed by the CPU 1101 when the host system 1100 writes the data of the disk cache 1103 into the optical disk 1201 is described with reference to FIGS. 1 and 2B. The embodiment to be described below is applied to the case in which the optical disk 1201 is an information recording medium in which data can be over-written.

Let it be supposed that the writing of the data of the front logical address (l) and the logical sector number (n) becomes necessary when the host system 1100 is reading out the program from the main memory 1102 into execution.

First, the CPU 1101 transfers the data to be written into the main memory 1102 to the disk cache 1103. In the UNIX which executes a data access in units of 512 bytes, the data address to be written into the information recording medium is designated by front logical address (l) and the number of blocks in the information recording medium is designated by the logical sector number (n).

In writing data, the CPU 1101 executes the following processing as shown in FIG. 2B.

(1) Calculation of front physical address (L) from front logical address (l):

If the front logical address (l) is an even number,

$$L=l/2$$

If the front logical address (l) is an odd number,

$$L=(l-1)/2$$

(2) Data read-out of front physical address (L) from optical disk 1201 when front logical address (l) is an odd number:

The CPU 1101 drives the interface circuit 1104 and outputs an instruction of reading out data of one sector from the front physical address (L) to the optical disk device 1200, reads the data of one physical sector from the front physical address (L) of the optical disk 1201, and stores this data in the converting buffer memory 1105 from "0" address thereof.

(3) Calculation of physical sector number (N) from logical sector number (n):

If the front logical address (l) and the logical sector number (n) are both even numbers;

$$N=n/2$$

If the front logical address (l) is an even number and the logical sector number (n) is an odd number;

$$N=(n+1)/2$$

If the front logical address (l) is an odd number and the logical sector number (n) is an even number;

$$N=(n/2)+1$$

If the front logical address (l) and the logical sector number (n) are both odd numbers;

$$N=(n+1)/2$$

(4) Calculation of final logical address (m) and final physical address (M):

The final logical address (m) is calculated as follows from the front logical address (l) and logical sector number (n):

$$m=l+n-1$$

If the final logical address (m) is an even number, the final physical address (M) is:

$$M=m/2$$

(5) Read-out of sectors of final physical address (M) from optical disk 1201 to sector when final logical address (m) is an even number:

If the final logical address (m) is an even number, first, an address (X) of the converting buffer memory 1105 in which the final physical sector is to be stored is calculated as follows based on the physical sector number (N):

$$X=1024 \times (N-1)$$

Next, an instruction of reading out the data of one sector from the final physical address (M) is outputted to the optical disk device 1200, data of one physical sector is read from the final physical address (M) of the optical disk 1201, and this data is stored in the converting buffer memory 1105 from the address (X) in which the final logical sector is to be stored.

(6) Overwriting of data of disk cache 1103:

The CPU overwrites the data to be written from the disk cache 1103 into the converting buffer memory 1105. If data exists on the converting buffer memory 1105 at this time, data is overwritten.

If the front logical address (l) is an even number, data having a capacity corresponding to the logical sector of the logical sector number (n) is written in the converting buffer memory 1105 from the front address thereof.

If the front logical address (l) is an odd number, data having a capacity corresponding to the logical sector of the logical sector number (n) is written in the converting buffer memory 1105 from "512" address thereof. "512" address is obtained by adding an offset corresponding to a capacity of one logical sector to "0" address.

(7) writing of data of converting buffer memory 1105 into optical disk 1201:

The CPU 1101 drives the interface circuit 1104 to output an instruction of writing data of the physical sector number (N) from the front physical address (L) to the optical disk device 1200 and transfers the data having a capacity corresponding to the physical sectors of the physical sector number (N) from "0" address of the converting buffer memory 1105 to the optical disk device 1200. In the optical disk device 1200, this data is written into the optical disk 1201 in the order from the front physical address (L) thereof.

## 11

The content of data to be stored in the converting buffer memory 1105 in reading out data from the optical disk 1201 is described below.

If the front logical address (l) and the logical sector number (n) are both even numbers, data is stored in the converting buffer memory 1105 as shown in FIG. 3B (a).

Data of the front logical address (l) and the logical sector number (n) transferred from the disk cache 1103 is stored in the converting buffer memory 1105 in the order from "0" address thereof. Data having a capacity corresponding to the physical sector of the physical sector number (N) is written into the optical disk 1201 in the order from the front physical address (L).

If the front logical address (l) is an even number and the logical sector number (n) is an odd number, data is stored in the converting buffer memory 1105 as shown in FIG. 3B (b).

Since the final logical address (m) is an even number, data of the physical sector of the final physical address (M) of the optical disk 1201 is read into the converting buffer memory 1105 in the order from the (X) address thereof. Then, data of the front logical address (l) and the logical sector number (n) transferred from the disk cache 1103 is stored in the converting buffer memory 1105 from "0" address thereof. At this time, the front half of 512 bytes of the data of the final physical address (M) already read is overwritten. Stored data corresponding to the physical sector number (N) is written into the optical disk 1201 from the front physical address (L) thereof.

If the front logical address (l) is an odd number and the logical sector number (n) is an even number, data is stored in the converting buffer memory 1105 as shown in FIG. 3B (c).

The data of the physical sector of the front physical address (L) of the optical disk 1201 is read into the converting buffer memory 1105 from "0" address thereof. Since the final logical address (m) is an even number, the data of the physical sector of the final physical address (M) of the optical disk 1201 is read into the converting buffer memory 1105 in the order from (X) address thereof. Thereafter, the front logical address (l) and the data of the logical sector number (n) transferred from the converting buffer memory 1105 are stored in "512" address thereof. At this time, the 512 bytes of the second half of the data of the front physical address (L) and 512 bytes of the first half of the data of the final physical address (M) already read are overwritten. All stored data of a capacity corresponding to the physical sector number (N) are written into the optical disk 1201 in the order from the front physical address (L).

If the front logical address (l) and the logical sector number (n) are both odd numbers, data is stored in the converting buffer memory 1105 as shown in FIG. 3B (d).

The data of the physical sector of the front physical address (L) of the optical disk 1201 is read into the converting buffer memory 1105 from "0" address thereof. In addition, the front logical address (l) and the data of the logical sector number (n) transferred from the disk cache 1103 are stored in the converting buffer memory 1105 in the order from "512" address thereof. At this time, 512 bytes of the second half of the data of the front physical address already read (L) are overwritten. All stored data of a capacity corresponding to the physical sector number (N) are written into the optical disk 1201 in the order from the front physical address (L).

As described above, according to the present invention, a request for data writing is converted, data is read from the optical disk 1201 into the converting buffer memory 1105 as necessary, data transferred from the disk cache 1103 is overwritten on data written into the converting buffer memory 1105, and the data on the converting buffer memory

## 12

1105 is written into the optical disk 1201. Thus, it is possible to use the optical disk 1201 having a sector size of 1024 bytes under an operating system in which data is written in units of 512 bytes.

According to the embodiment, the optical disk having the sector size of 1024 bytes can be used without damaging its capacity under the UNIX making a data access in units of 512 bytes.

The above description is concerned with the recording and reproduction of information with respect to an optical disk, but the present invention can be applied to an information recording medium other than the optical disk having a unit of recording and reproducing fixed data.

The present invention for safely using the optical disk which is a removable information recording medium under the UNIX which writes data in write-back method is described with reference to FIGS. 4, 5, 6, and 7.

FIG. 4 shows respective regions of the optical disk 1201 according to an embodiment of the present invention. The optical disk 1201 is divided into a disk label 4000, an (a) partition 4100, a (b) partition 4200, and a (c) partition 4300. The disk label 4000 comprises a disk identification information 4010, a UNIX control information 4020, and an open flag table 4030. The disk identification information 4010 is information for discriminating one optical disk from the other. When the construction of the UNIX is formed on an optical disk, a user allots a different value to the disk identification information 4010 for each optical disk. The UNIX control information 4020 includes information necessary for the UNIX to manage the optical disk 1201 such as information showing the position and size of each partition, a bit map showing the use state of each partition and the like. The open flag table 4030 comprises open flags 4031, 4032, and 4033 of each partition (a), (b), and (c). Each open flag takes a value either "in use" or "non-use".

FIGS. 5A, 5B, 5C, and 5D are flowcharts showing the flow of processing in controlling a removable medium according to the method for recording/reproducing information of the present invention.

FIG. 5A shows the flow of a control to be made when a partition is opening.

At step 5100, the CPU 1101 starts open processing according to the necessity of a data input to a partition and a data output therefrom. At step 5101, the CPU 1101 reads out the disk label 4000 from the optical disk 1201 and stores it in the disk label storing memory 1106. Then, at step 5102, the CPU 1101 obtains the UNIX control information 4020 from the disk label 4000. At step 5103, the CPU 1101 decides the mode of "open" of the relevant partition.

If it is decided that the mode of "open" is read-only, the CPU 1101 decides as to the condition of each partition of the open table 1108 at step 5108. The CPU 1101 prohibits the removal of the optical disk 1201 from the optical disk device 1200 only when all the partitions of the open table 1108 is "close" state at step 5109 and sets the state of the relevant partition of the open table 1108 to "open" at step 5110, and the CPU 1101 terminates the processing at step 5111 normally, i.e., "open" has succeeded.

If it is decided at step 5103 that "open" is in read/write mode, the CPU 1101 decides as to the type of the medium of the optical disk 1201 at step 5104. If the optical disk 1201 is an ROM medium, the CPU 1101 outputs an error message "Read-only file system" at step 5113, and at step 5114, the CPU 1101 terminates the processing as abnormal, i.e., "open" is unsuccessful.

## 13

If it is decided at step 5104 that the optical disk 1201 is a medium into which data can be written, the CPU 1101 decides as to the set state of the write protecting button at step 5105. If the write protecting button is set on the write inhibition side, the CPU 1101 outputs an error message "Read-only file system" at step 5113, thus terminating the processing as abnormal at step 5114, i.e., "open" is unsuccessful.

If the write protecting button is set on the write permission side, the CPU 1101 decides as to the state of the open flag of the relevant partition at step 5106. If the open flag of the relevant partition takes a value "in use", the CPU 1101 outputs an error message "Data inconsistency in partition ?" (? means a relevant partition) at step 5112, and outputs another error message "Read-only file system" at step 5113, thus terminating the processing as abnormal at step 5114, i.e., "open" is unsuccessful.

If it is decided at step 5106 that the open flag takes a value "non-use", the CPU 1101 sets the open flag of the relevant partition of the optical disk 1201 to "in use" at step 5107. Then, the CPU 1101 decides as to the state of each partition of the open table 1108 at step 5108. If each partition of the open table 1108 is in "close" state, the CPU 1101 inhibits the removal of the optical disk 1201 from the optical disk device 1200 at step 5109 and sets the state of the relevant partition of the open table 1108 to "open" at step 5110, thus terminating the processing as normal at step 5111, i.e., "open" is successful.

FIG. 5B shows the flow of the control when the CPU 1101 has detected that there is a possibility of replacing the optical disk 1201.

When the CPU 1101 has detected that there is a possibility of replacing the optical disk 1201 due to the turn-on of the power supply and so on, it starts control processing of replacing the disk at step 5200. The CPU 1101 decides as to the state of the open table 1108. If the state of all partitions of the open table 1108 is "close", the CPU 1101 terminates the processing at step 5206.

If the state of any one of the partitions of the open table 1108 is "open", the CPU 1101 reads out the disk label 4000 from the optical disk 1201 at step 5202. Then, the CPU 1101 compares with each other the disk identification information 4010 of the disk label 4000 thus read out and the disk identification information of the disk label already stored in the disk label storing memory 1106 at step 5203. If it is decided that the former is the same as the latter, the CPU 1101 issues an instruction of inhibiting the removal of the optical disk 1201 from the optical disk device 1200. If it is decided that the former is not the latter, the CPU 1101 sets the access prohibiting flag 1109 to "access prohibition" at step 5205. Then, the CPU 1101 does not communicate with the optical disk 1201 until the access prohibiting flag 1109 is released. Thus, the CPU 1101 terminates the processing for controlling the replacement of the disk at step 5206.

FIG. 5C shows the flow of the control to be made when the partition is "closing".

The CPU 1101 starts close processing at step 5300 when a data input to the relevant partition and a data output therefrom become unnecessary. First, the CPU 1101 decides as to the state of the access prohibiting flag 1109 at step 5301. When the access prohibiting flag 1109 is set to "access prohibition", the CPU 1101 sets the state of the relevant partition of the open table 1108 to "close" at step 5305. Then, at step 5306, the CPU 1101 decides as to the state of each partition of the open table 1108. If it is decided that the state of each partition of the open table 1108 is "close", the CPU 1101 permits the removal of the optical disk 1201 from

## 14

the optical disk device 1200 at step 5307. Then, at step 5308, the CPU 1101 sets the state of the access prohibiting flag 1109 to "access permission". At step 5308, the CPU 1101 sets the state of the access prohibiting flag 1109 to "access permission". If the access prohibiting flag 1109 is already in the state of "access permission", the processing executed at step 5308 has no particular meaning. Thus, the CPU 1101 terminates the close processing at step 5309.

If it is decided at step 5301 that the access prohibiting flag 1109 is set to "access permission", the CPU 1101 writes data into the optical disk 1201 at step 5302 if this data to be written into the relevant partition is on the disk cache 1103. If the relevant partition is opened in read-only mode, there is no data to be written to the optical disk 1201. Therefore, this processing is skipped. At step 5303, the CPU 1101 decides as to the mode of "open". Only when the relevant partition is opened in read/write mode, the CPU 1101 sets the open flag of the relevant partition of the disk label to the state of "non-use" and writes data into the disk label 4000 of the optical disk 1201 at step 5304.

Then, the CPU 1101 sets the state of the relevant partition of the open table 1108 to "close" at step 5305. The CPU 1101 then decides as to the state of each partition of the open table 1108 at step 5306. If only all the partitions of the open table 1108 are in the state of "close", the CPU 1101 permits the removal of the optical disk 1201 from the optical disk device 1200 at step 5307, and sets the state of the access prohibiting flag 1109 to "access permission" at step 5308. Then, the CPU 1101 terminates the close processing at step 5309.

FIG. 5D shows the flow of the control over the partition in correcting the inconsistency of the file system.

On receipt of an instruction from a user, the CPU 1101 starts the processing for correcting the inconsistency of the file system at step 5400. The CPU 1101 reads out the disk label 4000 from the optical disk 1201, stores it in the main memory 1102 at step 5401, and obtains the UNIX control information 4020 from the disk label 4000 at step 5402.

Then, the CPU 1101 reads out the control information of the file system from the relevant partition of the optical disk 1201 to investigate if there is inconsistency. If there is inconsistency, the CPU 1101 corrects the inconsistency. At step 5404, the CPU 1101 decides as to the state of the open flag of the relevant partition of the disk label 4000 stored in the main memory 1102. If it is decided that the open flag is set to "in use", the CPU 1101 sets the open flag of the relevant partition of the disk label 4000 stored in the main memory 1102 to "non-use" and writes it into the optical disk 1201 at step 5405, thus terminating the processing at step 5406. If it is decided that the open flag is set to "non-use", the CPU 1101 terminates the processing at step 5406.

An embodiment is described below more specifically by way of examples of operation. FIG. 6 shows the content of the open table 1108 according to an embodiment of the present invention. FIG. 7 shows the content of the open flag table 4030 according to an embodiment of the present invention. The conversion processing of sector size previously described is always executed in the input/output of data. But the description of the conversion processing of sector size is omitted.

Description is made below by way of an example in which the optical disk 1201 is a writable information recording medium and the write protecting button is set on the writable side.

First, in the host system 1100, none of the partitions are used. As shown in FIG. 6 (a), the states of all the partitions of the open table 1108 are "close". As shown in FIG. 7 (a), all the partitions of the open flag table 4030 are "non-use". The access prohibiting flag 1109 is set to the state of "access permission".



First, the partition to be used is opened.

Let it be supposed that the writing of data into the (a) partition 4100 of the optical disk 1201 is necessary when the CPU 1101 is reading out the program from the main memory 1102 into execution. The CPU 1101 starts the open processing in the read/write mode of the (a) partition at step 5100. The CPU 1101 reads out the disk label 4000 from the optical disk 1201, stores it in the disk label storing memory 1106 at step 5101, and obtains the UNIX control information 4020 from the disk label 4000.

Since the partition is opened at step 5103 in the read/write mode, the CPU 1101 decides as to the type of the medium of the optical disk 1201 at step 5104. The CPU 1101 issues a Mode Sense command of SCSI to the optical disk device 1200 via the interface circuit 1104. In response to the command, the optical disk device 1200 produces data including the set state of the write protecting button of the optical disk 1201 and the type of the medium of the optical disk 1201, thus transferring the data to the host system 1100. The CPU 1101 stores this data in the disk state register 1107 and takes out the type of the medium from this data.

Since the optical disk 1201 is a writable medium, the CPU 1101 decides as to the set state of the write protecting button at step 5105. The CPU 1101 takes out the set state of the write protecting button from the disk state register 1107.

Since the write protecting button is set on the writable side, the CPU 1101 decides as to the state of the open flag 4031 of the (a) partition at step 5106. The CPU 1101 takes out the open flag 4031 of the (a) partition from the disk label storing memory 1106 storing the disk label 4000.

The open flag table 4030 is as shown in FIG. 7(a) and the open flag 4031 of the (a) partition is set to "non-use", so that the CPU 1101 sets the open flag 4031 of the (a) partition on the optical disk 1201 to "in use" at step 5107. The CPU 1101 sets the open flag 4031 of the (a) partition of the disk label 4000 stored in the disk label storing memory 1106 to "in use" and writes it into the optical disk 1201 as recording data. At this time, the open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(b).

Then, the CPU 1101 decides as to the state of each partition of the open table 1108 at step 5108. Since the states of all the partitions are in the state of "close" as shown in FIG. 6(a), the CPU 1101 prohibits the removal of the optical disk 1201 from the optical disk device 1200 at step 5109. More specifically, the CPU 1101 issues a Prevent Medium Removal command of SCSI to the optical disk device 1200 via the interface circuit 1104. In response to this command, the optical disk device 1200 prohibits the removal of the optical disk 1201 from the optical disk device 1200. The removal prohibition of the optical disk 1201 from the optical disk device 1200 is not released until the optical disk device 1200 receives Allow Medium Removal command of SCSI from the host system or the optical disk device 1200 is reset.

The CPU 1101 sets the state corresponding to the (a) partition 4100 in the open table 1108 to "open" at step 5110 as shown in FIG. 6(b), thus terminating the "open" processing at step 5111.

Then, the CPU 1101 writes data. The CPU 1101 transfers data from the main memory 1102 to the disk cache 1103. The CPU 1101 records the data of the disk cache 1103 into the optical disk 1201 of the optical disk device 1200 by using the write-back method. The CPU 1101 transfers data to the disk cache 1103 when a request for the data writing of the program in execution is made, but does not write data into the optical disk 1201. Data is written into the optical disk 1201 after a certain period of time elapses. The optical disk device 1200 is in the removal prohibition state of the optical

disk 1201 at this time, the optical disk 1201 is not removed even though the user makes an erroneous request for the removal of the optical disk 1201 from the optical disk device 1200 to the optical disk device 1200.

Subsequently, another partition is opened.

Let it be supposed that data writing into the (b) partition 4200 is necessary when the CPU 1101 is reading out another program from the main memory 1102 into execution. The open processing of the (b) partition 4200 is performed similarly to the (a) partition 4100. The CPU 1101 starts the "open" processing under the read/write mode of the (b) partition at step 5100. Then, the CPU 1101 reads out the disk label 4000, stores it in the disk label storing memory 1106 at step 5101, obtains the UNIX control information 4020 at step 5102, decides as to the mode of "open" at step 5103, makes a decision about the type of the medium at step 5104, and makes a decision as to the set state of the write protecting button at step 5105.

Then, the CPU 1101 takes out the open flag 4032 of the (b) partition from the disk label storing memory 1106, thus deciding as to this state at step 5106. Since the open flag table 4030 is as shown in FIG. 7(b) and the open flag 4032 of the (b) partition is set to "non-use", the CPU 1101 sets the open flag 4032 of the (b) partition on the optical disk 1201 to "in use" at step 5107. At this time, the open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(b).

Then, the CPU 1101 decides as to the state of the open table 1108 at step 5108. The open table is "open" when the open table has a state corresponding to the (a) partition 4100. Since the CPU 1101 detects that the optical disk device 1200 is already in the removal prohibition state of the optical disk 1201, the CPU 1101 sets the state corresponding to the (b) partition 4200 of the open table 1108 to "open" at step 5100, thus terminating the open processing at step 5111.

Then, the CPU 1101 transfers data, to be written into the (b) partition, from the main memory 1102 to the disk cache 1103. At this time, the optical disk 1201 is prohibited from being removed from the optical disk device 1200.

Next, the (a) partition 4100 is closed. The CPU 1101 starts "close" processing of the (a) partition 4100 at step 5300. The CPU 1101 makes a decision as to the state of the access prohibiting flag 1109 at step 5301. Since the access prohibiting flag 1109 is in the state of "access permission", the CPU 1101 writes all data, to be written into the 4100, into the (a) partition of the optical disk 1201 to be accommodated in the optical disk device 1200 by driving the interface circuit 1104 at step 5302.

Then, the CPU 1101 makes a decision as to the mode of "open" of the (a) partition 4100. Since the (a) partition 4100 is opened in the read/write mode, the CPU 1101 sets the open flag 4031 of the (a) partition of the optical disk 1201 to "non-use" at step 5304. More specifically, the CPU 1101 sets the open flag 4031 of the (a) partition of the disk label 4000 already stored in the disk label storing memory 1106 to "non-use" and writes it into the optical disk 1201 accommodated in the optical disk device 1200 as recording data. At this time, the open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(d).

Then, the CPU 1101 sets the state of the (a) partition 4100 of the open table 1108 to the state of "close" as shown in FIG. 6(d) at step 5305. Thereafter, the CPU 1101 makes a decision about the state of each partition of the open table 1108 at step 5306. Since the state of the (b) partition is "open" and as such "in use" as shown in FIG. 6(d), the CPU 1101 terminates the "close" processing at step 5309.

The optical disk 1201 is prohibited from being removed from the optical disk device 1200. Therefore, if the user erroneously makes a request for the removal of the optical disk 1201 to the optical disk device 1200, the optical disk 1201 is not removed.



Let it be supposed that the user turns off the power source of the optical disk device 1200 and then, turns it on. Since the optical disk device 1200 is reset, the removal prohibition of the optical disk 1201 is released.

Since data to be written exists in the disk cache 1103 for a certain period of time, the CPU 1101 issues a WRITE command of SCSI to the optical disk device 1200 via the interface circuit 1104 in order to write this data into the (b) partition 4200 of the optical disk 1201. In response to this command, the optical disk device 1200 reports the Unit Attention state by the resetting of the optical disk device 1200 itself. In response to the report, the CPU 1101 of the host system 1100 starts processing for controlling the replacement of the optical disk at step 5200.

The CPU 1101 makes a decision as to the state of the open table 1108 at step 5201. Since the open table 1108 has a partition which is not in "close" state as shown in FIG. 6(d) and as such the optical disk 1201 is in use, the CPU 1101 reads out the disk label 4000 from the optical disk 1201 and stores it in the main memory 1102 at step 5202.

Thereafter, the CPU 1101 compares with each other the disk identification information 4010 of the disk label 4000 read out and disk identification information of the disk label stored in the disk label storing memory 1106 when the (b) partition 4200 is opened in order to decide whether the former is the same as the latter at step 5203. Since the optical disk 1201 has not been replaced and as such the former is the same as the latter, the CPU 1101 issues to the optical disk device 1200 an instruction indicative of the removal prohibition of the optical disk 1201 at step 5204. Thus, the CPU 1101 terminates the processing for controlling the disk replacement at step 5206.

As described above, since the optical disk device 1200 is set again to the removal prohibition state of the optical disk 1201, the optical disk 1201 is not removal even though the user erroneously makes a request for the removal thereof.

The CPU 1101 writes data into the (b) partition 4200 of the optical disk 1201 after terminating the processing for controlling the disk replacement.

Finally, the (b) partition 4200 is closed. The "close" processing of the (b) partition 4200 is executed similarly to the (a) partition 4100. The CPU 1101 starts "close" processing upon completion of the program at step 5300. Subsequently, the CPU 1101 makes a decision as to the state of the access prohibiting flag 1109 at step 5301, writes data in the disk cache 1103 into the (b) partition 4200 at step 5302, makes a decision as the mode of "open" of the (b) partition 4200 at step 5303, and sets the open flag 4032 of the (b) partition to "non-use" at step 5304. At this time, the open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(e).

Then, the CPU 1101 sets the state of the (b) partition 4200 of the open table 1108 to "close" at step 5303 and makes a decision as to the state of each partition of the open table 1108 at step 5306. At this time, all the partitions of the open table 1108 are in "close" state as shown in FIG. 6(e) and therefore, it is understood that none of the partitions of the optical disk 1201 cannot be used. Accordingly, the CPU 1101 issues a command of Allow Medium Removal of SCSI to the optical disk device 1200 to permit the removal of the optical disk 1201 from the optical disk device 1200 at step 5307. In response to this command, the optical disk device 1200 enables the user to take out the optical disk 1201 thereafter.

Then, the CPU 1101 resets the access prohibiting flag 1109 to "access permission" state at step 5308, thus terminating the "close" processing at step 5309.

In this state, the user can take out the optical disk 1201 from the optical disk device 1200. At this time, all partitions of the optical disk 1201 are closed and all data of the disk cache 1103 have been already written into the optical disk 1201. Therefore, no problems occur even though the optical disk 1201 is taken out. In addition, if the optical disk 1201 is replaced to another since all the partitions of the optical disk 1201 have been already closed, the data of the previously accommodated optical disk 1201 is not written into the currently accommodated optical disk 1201.

Data is written again into the (a) partition of the optical disk 1201.

The (a) partition is opened similarly to the above-described example. When the "open" processing of the (a) partition terminates, the open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(f). At this time, the state of the open table 1108 is as shown in FIG. 6(f).

The CPU 1101 writes data by using write-back method. The CPU 1101 transfers data to the disk cache 1103 when there is a request for data writing of a program in execution, but does not write data into the optical disk 1201.

The user turns off the power source of the optical disk device 1200 and then turns it on again. The optical disk device 1200 is reset and the removal prohibition of the optical disk 1201 is released. The user takes out the optical disk 1201 and puts a different optical disk (not shown) into the optical disk device 1200. At this time, there is data inconsistency in the file system of the (a) partition 4100 of the optical disk 1201 taken out because data which should be written has not been written. The open flag 4031 of the optical disk 1201 remains "in use".

The CPU 1101 issues a Write command of SCSI to the optical disk device 1200 to write data into the (a) partition of the optical disk 1201. In response to this command, the optical disk device 1200 reports Unit Attention state brought about by resetting. In response to this report, the CPU 1101 starts the processing for controlling a disk replacement at step 5200.

The CPU 1101 makes a decision as to the state of the open table 1108 at step 5201. At this time, the open table 1108 has a partition which is not in "close" state as shown in FIG. 6(f) and as such the optical disk 1201 is in use. Therefore, the CPU 1101 reads out the disk label 4000 from the optical disk 1201 and stores it in the main memory 1102 at step 5202.

The CPU 1101 decides at step 5203 as to whether or not the disk identification information 4010 of the disk label 4000 is the same as the disk identification information of the disk label stored in the disk storing memory 1106. Since both are different from each other, it is understood that an optical disk accommodated in the optical disk device 1200 is different from the optical disk 1201 which has been already accommodated in the optical disk device 1200. Therefore, the CPU 1101 sets the access prohibiting flag 1109 to "access prohibition" at step 5205. Thus, the CPU 1101 terminates the processing for controlling the disk replacement at step 5206.

The CPU 1101 does not communicate with the optical disk 1201 when the access prohibiting flag 1109 is set to "access prohibition".

Accordingly, even though the optical disk 1201 is forcibly replaced with a different optical disk when the partition of the optical disk 1201 is being used by opening it, the data which should have been written into the previously accommodated optical disk is not written into the optical disk 1201 accommodated currently.

Finally, the partition used is closed. Upon termination of the program, the CPU 1101 starts the "close" processing of the (a) partition 4100 at step 5300. Then, the CPU 1101 makes a decision as to the state of the access prohibiting flag 1109 at step 5301. Since the access prohibiting flag 1109 is set to "access prohibition", the CPU 1101 sets the state of the (a) partition 4100 of the open table 1108 to "close" at step 5305 and makes a decision as to the state of each partition of the open table 1108 at step 5306. At this time, as shown in FIG. 6(g), all the partitions of the open table 1108 are in "close" state. Then, the CPU 1101 issues an instruction indicating of the removal permission of the optical disk 1201 to the optical disk device 1200 at step 5307. But since the optical disk device 1200 is already in the removal permission state, this operation has no particular meaning. Then, the CPU 1101 sets the access prohibiting flag 1109 "access permission" at step 5308, thus terminating the "close" processing at step 5309.

The case in which the (a) partition 4100 of the optical disk 1201 opened by the host system 1100 and taken out during use is opened again to use it in read/write mode is described below.

The CPU 1101 starts an "open" processing in the read/write mode of the (a) partition 4100 at step 5100. The CPU 1101 reads out the disk label 4000, stores it in the disk storing memory 1106 at step 5101, obtains the UNIX control information 4020 at step 5102, makes a decision as to the mode of "open" at step 5103, makes a decision as to the type of the medium at step 5104, and makes a decision as to the set state of the write protecting button at step 5105.

Then, the CPU 1101 takes out the open flag 4031 of the (a) partition from the disk storing memory 1106 and decides as to the state thereof at step 5106. But in this optical disk 1201, the (a) partition 4100 is not normally closed. Therefore, the open flag 4031 of the (a) partition of the open flag table 4030 remains "in use" as shown in FIG. 7(f). This indicates that the inconsistency of the file system exists in the (a) partition 4100. Thus, the CPU 1101 indicates error messages of "Data inconsistency in partition A" and "Read-only file system" at steps 5112 and 5113, thus terminating the "open" processing as abnormal at step 5114.

Therefore, the (a) partition 4100 with the file system having an inconsistency is not opened in the read/write mode and an existing file is not destroyed or the system does not go down.

Then, the file system of the (a) partition 4100 of the optical disk 1201 is corrected.

The user knows the occurrence of the inconsistency of the file system because of the failure of "open" and upon request from the user, the CPU 1101 executes the program in order to correct the file system.

The CPU 1101 starts the correction processing of the file system at step 5400. The CPU 1101 reads out the disk label 4000 from the optical disk 1201, stores it in the disk storing memory 1106 at step 5401, and obtains the UNIX control information 4020 including the map information of the partition at step 5402.

Then, the CPU 1101 checks whether or not the file system of the (a) partition 4100 of the optical disk 1201 has inconsistencies. If inconsistencies is detected, the (a) partition 4100 is corrected into a consistent state at step 5403.

Then, the CPU 1101 decides as to the state of the open flag 4031 of the (a) partition at step 5404. As shown in FIG. 7(f), since the open flag 4031 of the (a) partition is set to "in use", the CPU 1101 sets the open flag 4031 of the (a) partition of the disk label 4000 stored in the disk storing memory 1106 to "non-use" and writes the disk label 4000 into the optical

disk 1201 at step 5405, thus terminating the processing at step 5406.

The open flag table 4030 of the optical disk 1201 is as shown in FIG. 7(g) at this time and the open flag 4031 of the (a) partition is set to "non-use". Therefore, the (a) partition 4100 can be used by opening it again in the read/write mode. Since the inconsistency of the file system of the (a) partition 4100 has been already corrected, the existing file is not destroyed or the system does not go down due to the writing of new data.

Next, the case in which the optical disk 1201 is a writable information recording medium and the write protecting button is set on the write prohibiting side is described below.

Let it be supposed that the CPU 1101 is going to write data into the (a) partition 4100.

The CPU 1101 starts the "open" processing in the read/write mode of the (a) partition 4100 at step 5100. Then, the CPU 1101 reads out the disk label 4000 and stores it in the disk label storing memory 1106 at step 5101, obtains the UNIX control information 4020 at step 5102, and decides as to the mode of "open" at step 5103. Then, the CPU 1101 decides as to the type of the medium at step 5104. Since the optical disk 1201 is a writable medium, the CPU 1101 decides as to the set state of the write protecting button at step 5105. Since the write protecting button is set on the write prohibiting side, the CPU 1101 displays an error message "Read-only file system" to the user at step 5113, thus terminating the "open" processing as abnormal at step 5114.

If the optical disk 1201 is an ROM medium and data cannot be written in the above example, the "open" processing is executed as follows: When the CPU 1101 makes a decision as to the type of the medium at step 5104, and displays an error message "Read-only file system" to the user at step 5113, thus terminating the "open" processing at step 5114.

As described above, the partition of the optical disk 1201 into which data cannot be written is not opened in the read/write mode. Therefore, the system does not go due to mismatch between data of the optical disk 1201 and that of the disk cache 1103.

As apparent from the foregoing description, the removal of the optical disk 1201 from the optical disk device 1200 is controlled according to the open state of the partition, which prevents the destruction of data.

According to the embodiment of the present invention, upon detection of the possibility of the replacement of the optical disk 1201 when the partition has been already opened, whether or not the optical disk 1201 has been replaced is decided by using the disk identification information 4010. If it is decided that the optical disk 1201 has not been replaced, processing continues with the optical disk 1201 prohibited from being removed from the optical disk device 1200. If it is decided that the optical disk 1201 has been replaced, an access to the optical disk accommodated in the optical disk device 1200 is prohibited. Thus, when the optical disk device 1200 is reset during use, the removal of the optical disk 1201 is prohibited again to prevent data from being destroyed. If the optical disk 1201 is forcibly replaced, an access to the replaced optical disk 1201 is prohibited to prevent the destruction of the data of the optical disk.

According to the embodiment of the present invention, the open flag of the optical disk 1201 corresponding to the partition is set to "in use" when the partition is opened and the open flag is set to "non-use" when the partition is closed. When the open flag has been set to "in use" when the partition is opened, the partition is prohibited from being

opened in the read/write mode. Thus, even though the optical disk 1201 is forcibly taken out from the optical disk device 1200 during use, data is prevented from being written into a file system having inconsistency. Therefore, existing file is not destroyed or the system does not go down.

Further, the inconsistency of the file system of the partition on the optical disk 1201 taken out during use is corrected and the open flag on the optical disk 1201 is set to "non-use". Thus, the partition is opened again in the read/write mode to safely use the apparatus.

In addition, the partition on the optical disk into which data cannot be written is prohibited from being opened in the read/write mode to prevent the system from going down because of a mismatch between the data on the optical disk and the data on the disk cache.

In the embodiment, the open flag table 4030 is on the disk label 4000, but an open flag corresponding to each partition can be anywhere on the optical disk except the user's access area.

In the embodiment, the user gives an instruction to the CPU 1101 so that the file system is corrected. In this case, the user is requested to perform an operation so that the inconsistency of the file system is corrected and a possible destruction of data due to turn-off of the power source is warned to the user. It is possible that the CPU 1101 automatically corrects the inconsistency of the file system when it has detected the inconsistency in "open" processing. Thus, the "open" processing can be terminated normally, which eliminates the user's operation.

The information recording/reproducing method and apparatus of the present invention can be applied to an apparatus using a removable information recording medium other than an optical disk in the controlling process.

In addition, the information recording/reproducing method and apparatus of the above-described embodiment of the present invention use a disk cache of write-back system, but the information recording/reproducing method and apparatus in which data writing is executed from "open" process until "close" processing.

Although the present invention has been fully described in connection with the preferred embodiments thereof with reference to the accompanying drawings, it is to be noted that various changes and modifications are apparent to those skilled in the art. Such changes and modifications are to be understood as included within the scope of the present invention as defined by the appended claims unless they depart therefrom.

What is claimed is:

1. A method for recording and reproducing information carried out by an apparatus employing an information recording medium having at least one partition composed of a plurality of files and file management information and a flag area in which an open flag is recorded in one-to-one correspondence with each partition, said open flag setting an in-use state in a partition where logically imperfect file management information is recorded and setting a non-use state in a partition where logically perfect file management information is recorded, said method having a partition opening process conducted before the start of a file record-

ing operation and a partition closing process conducted after the completion of the file recording operation,

said partition opening process comprising the steps of: discriminating the states of the open flags read out from the flag area;

error-terminating said opening process when at least one of the open flags shows the in-use state is detected in said flag discriminating step; and

setting the open flag corresponding to the partition to be subjected to the partition opening process to the in-use state when only the open flag shows the non-use state is detected in said flag discriminating step and recording on said flag area,

said partition closing process comprising the steps of setting the open flag corresponding to the partition to be subjected to the closing process to the non-use state and recording on said flag area.

2. A method as defined in claim 1, further including a file system restoring process for restoring the file management information in the partition where a logical inconsistency is detected, said file system restoring processing comprising a step of setting the open flag corresponding to the partition where the logic inconsistency is detected to the non-use state and recording on said flag area.

3. An apparatus for recording and reproducing information employing an information recording medium having at least one partition composed of a plurality of files and file management information and a flag area in which an open flag is recorded in one-to-one correspondence with each partition, said open flag setting an in-use state in a partition where logically imperfect file management information is recorded and setting a non-use state in a partition where logically perfect file management information is recorded, said apparatus comprising:

an opening means for conducting the partition opening processing before the start of the file recording and reproducing operation, said partition opening means comprising:

- (a) flag discriminating means for discriminating the state of open flags read out from the flag area;
- (b) means for error-terminating the partition opening process when said flag discriminating means detects at least one of the open flags showing the in-use state; and

- (c) means for setting the open flag corresponding to the partition to be subjected to the partition opening process to the in-use state when said flag discriminating means detects only the open flag showing the non-use state and recording on said flag area; and

a closing means for conducting the partition closing process after completion of the file recording and reproducing operation, said partition closing means comprising:

- means for setting the open flag corresponding to the partition to be subjected to the closing process to the non-use state and recording on said flag area.

\* \* \* \* \*